# Przedmowa

Książka opisuje te wyniki projektu "Inżynieria Internetu Przyszłości", które zakończyły się działającymi prototypami i zostały zademonstrowane na wystawach krajowych i międzynarodowych.

Książka zaczyna się od opisu Systemu IIP poziom 1 i 2 (zgodnie z założoną architekturą opisaną w części 1), który odpowiada za utworzenie wirtualnej infrastruktury sieciowej, pozwalającej na zbudowanie wirtualnych węzłów i łączy. Ważnym elementem tej infrastruktury jest zapewnienie izolacji pomiędzy poszczególnymi wirtualnymi węzłami/łączami należącymi do różnych Równoległych Internetów, pomimo iż są one ustanowione na tych samych fizycznych węzłach umożliwiających wirtualizację (EZappliance, Xen i NetFPGA) i fizycznych łączach. Ponadto, przedstawiono sposób wymiarowania zasobów współdzielonej infrastruktury pomiędzy Równoległe Internety oraz podsystem zapewnienia bezpieczeństwa sieci.

W następnych rozdziałach przedstawiono szczegóły dotyczące dwóch Równoległych Internetów, tj. IPv6 QoS oraz Sieci Świadomej Przekazywanej Treści. Internety te korzystają z tej samej infrastruktury fizycznej, ale działają w oparciu o wirtualne węzły i wirtualne łącza. Internety te różnią się pomiędzy sobą zastosowanymi rozwiązaniami dotyczącymi warstwy sterowania i warstwy przekazu danych. W szczególności opisano: zasadę działania poszczególnych Internetów, zastosowaną w sieci sygnalizację, sposób przyłączenia aplikacji, metodę rezerwacji zasobów oraz wybrane wyniki przeprowadzanych testów.

Kolejnym ważnym wynikiem projektu jest sieć badawcza PL-LAB, która na dzień dzisiejszy łączy 8 ośrodków krajowych i pozwala na zdalne konfigurowanie eksperymentów i ich przeprowadzanie. Eksperymenty te głównie dotyczą wirtualizacji infrastruktury sieciowej.

Ostatnim tematem poruszanym w książce jest transformacja sieci z IPv4 do IPv6. W szczególności przedstawiono zagadnienia związane ze scenariuszami biznesowymi w środowisku IPv6, aspekty zarządzania mobilnością, dobre praktyki procesu migracji do wersji IPv6 oraz implementację protokołów SNMP, IPSec oraz DHCP.

Na zakończenie chciałbym serdecznie podziękować wszystkim tym, którzy swoją pracą przyczynili się do opracowania wspomnianych prototypów.

Wojciech Burakowski
Koordynator projektu "Inżynieria Internetu Przyszłości"

Badania, których wyniki są przedstawione w tej książce, są prowadzone w ramach projektu „Inżynieria Internetu Przyszłości" częściowo finansowanego przez Ministerstwo Nauki i Szkolnictwa Wyższego z Europejskiego Funduszu Rozwoju Regionalnego w ramach Programu Operacyjnego Innowacyjna Gospodarka na lata 2007-2013, umowa nr POIG.01.01.01-00-045/09.

# Wykonawcy projektu „Inżynieria Internetu Przyszłości”

**Politechnika Warszawska:**
prof. dr hab. inż. Wojciech Burakowski (kierownik projektu, kierownik *Systemu IIP*)
prof. dr hab. inż. Zbigniew Kotulski
prof. dr hab. inż. Michał Pióro
prof. dr hab. inż. Paweł Szczepański (kierownik *Światłowodowej Sieci Dostępowej*)
dr inż. Andrzej Bęben (zastępca kierownika projektu, kierownik *Sieci Świadomej Przekazywanej Treści*)
dr inż. Krzysztof Cabaj
dr inż. Michał Jarociński
dr inż. Mariusz Rawski (kierownik *Platformy NetFPGA*)
dr Paweł Szałachowski
dr inż. Piotr Szotkowski
dr inż. Halina Tarasiuk (kierownik *IPv6 QoS*)
dr inż. Paweł Tomaszewicz
dr inż. Piotr Zwierko
dr inż. Mateusz Żotkiewicz
mgr inż. Mateusz Bugaj
mgr inż. Mariusz Mycek
mgr inż. Katrin Welikow
mgr inż. Piotr Wiśniewski

**Politechnika Gdańska:**
prof. dr hab. inż. Józef Woźniak
dr hab. inż. Jerzy Konorski (kierownik *Bezpieczeństwa Sieci*)
dr inż. Tomasz Gierszewski
dr inż. Wojciech Gumiński
dr inż. Tomasz Mrugalski
dr inż. Krzysztof Nowicki
dr inż. Jacek Rak
mgr inż. Krzysztof Gierłowski
mgr inż. Michał Hoeft
mgr inż. Przemysław Machań
mgr inż. Karol Podolski
mgr inż. Mariusz Stankiewicz
mgr inż. Jacek Świątowiak

**Politechnika Wrocławska:**
prof. dr hab. inż. Adam Grzech (zastępca kierownika projektu, kierownik *Przyszłych Aplikacji*)
prof. dr hab. inż. Jerzy Świątek
dr inż. Krzysztof Brzostowski
dr inż. Krzysztof Chudzik
dr inż. Dariusz Gąsior
dr inż. Jarosław Drapała
dr inż. Krzysztof Juszczyszyn
dr inż. Grzegorz Kołaczek
dr inż. Andrzej Kozik
dr inż. Jan Kwiatkowski
dr inż. Radosław Rudek
dr inż. Paweł Świątek
mgr inż. Maciej Drwal
mgr inż. Jakub Forjasz
mgr inż. Kamil Nowak
inż. Maciej Demkiewicz
inż. Piotr Klukowski
inż. Adam Kokot
inż. Patryk Schauer
Paweł Starak

**Instytut Łączności – Państwowy Instytut Badawczy:**
dr inż. Paweł Białoń
dr inż. Janusz Granat (kierownik *Systemu Zarządzania*)
dr inż. Piotr Krawiec (kierownik *PL-LAB*)
dr inż. Jordi Mongay Batalla
mgr inż. Jakub Bielecki
mgr inż. Mariusz Gajewski

mgr inż. Henryk Gut
mgr inż. Waldemar Latoszek
mgr inż. Tomasz Lewicki
mgr inż. Paweł Olender
mgr inż. Jan Rogowski
mgr inż. Konrad Sienkiewicz
mgr inż. Maciej Sosnowski
inż. Wojciech Szymak
mgr inż. Piotr Skałecki
mgr inż. Eryk Skotarczak
mgr inż. Aleksander Stroiński
mgr inż. Maciej Stróżyk
mgr inż. Jakub Szubarga
mgr inż. Piotr Szymaniak
mgr inż. Dariusz Walczak
mgr inż. Marcin Wrzos

**Poznańskie Centrum Superkomputerowo-Sieciowe:**
dr inż. Ewa Kuśmierek
dr inż. Cezary Mazurek
dr inż. Maciej Stroiński
mgr inż. Bartosz Belter
mgr inż. Artur Binczewski (kierownik *Transformacji z IPv4 do IPv6*)
mgr inż. Wojbor Bogacki
mgr inż. Jakub Borzdyński
mgr inż. Robert Cecko
mgr inż. Łukasz Dolata
mgr inż. Krzysztof Dombek
mgr inż. Bartosz Gajda
mgr inż. Maciej Głowiak
mgr inż. Jakub Gutkowski
mgr inż. Bartłomiej Idzikowski
mgr inż. Jerzy Jamroży
mgr inż. Michał Kosiedowski
mgr inż. Artur Kryszak
mgr inż. Radosław Krzywania
mgr inż. Łukasz Lechert
mgr inż. Bartosz Lewandowski
mgr inż. Szymon Malewski
mgr inż. Tomasz Merda
mgr inż. Łukasz Ogrodowczyk
mgr inż. Iwo Olszewski
mgr inż. Piotr Ostapowicz
mgr inż. Damian Parniewicz (kierownik *Platformy EZappliance*)
mgr inż. Piotr Pawałowski
mgr inż. Wojciech Pieklik
mgr inż. Wiktor Procyk
mgr inż. Miłosz Przywecki
mgr inż. Marcin Radoń
mgr inż. Arkadiusz Radziuk

**Politechnika Poznańska:**
prof. dr hab. inż. Wojciech Kabaciński
prof. dr hab. inż. Maciej Stasiak
prof. dr hab. inż. Krzysztof Wesołowski
dr hab. inż. Grzegorz Danilewicz (kierownik *Sieci Komutacji Strumieni*)
dr hab. inż. Mariusz Głąbowski
dr inż. Sławomir Hanczewski
dr inż. Adam Kaliszan
dr inż. Janusz Kleban
dr inż. Robert Kotrys
dr inż. Rafał Krenz (kierownik *Kratowej Sieci Bezprzewodowej*)
dr inż. Marek Michalski
dr inż. Piotr Zwierzykowski
dr inż. Mariusz Żal
mgr inż. Krzysztof Bąkowski
mgr inż. Marcin Dziuba
mgr inż. Remigiusz Rajewski
mgr inż. Marcin Rodziewicz
mgr inż. Maciej Sobieraj

**Instytut Informatyki Teoretycznej i Stosowanej PAN:**
prof. dr hab. inż. Tadeusz Czachórski (kierownik *Rozpowszechniania Wyników*)
dr inż. Przemysław Głomb
dr inż. Mateusz Nowak
dr inż. Sławomir Nowak
dr inż. Piotr Pecka
dr inż. Arkadiusz Sochan
mgr Michał Cholewa
mgr inż. Sebastian Opozda

mgr inż. Michał Romaszewski

**AGH Akademia Górniczo-Hutnicza w Krakowie:**
prof. dr hab. inż. Andrzej Pach
prof. dr hab. inż. Zdzisław Papir
prof. dr hab. inż. Tomasz Zieliński
dr inż. Jarosław Bułat
dr inż. Marek Natkaniec (kierownik *Bezprzewodowej Sieci Dostępowej*)
dr inż. Janusz Gozdecki
dr inż. Michał Grega
dr inż. Lucjan Janowski
dr inż. Mirosław Kantor
dr inż. Jerzy Kasperek
dr inż. Katarzyna Kosek-Szott
dr inż. Mikołaj Leszczuk
dr inż. Piotr Pacyna
dr inż. Paweł Rajda
dr inż. Szymon Szott
dr inż. Andrzej Szymański
dr inż. Krzysztof Wajda
mgr inż. Dawid Juszka
mgr inż. Krzysztof Łoziak
mgr inż. Wojciech Romaszkan
mgr inż. Marcin Rupiński
mgr inż. Jacek Rząsa
mgr inż. Dominik Rzepka
mgr inż. Grzegorz Rzym
mgr inż. Marek Sikora
mgr inż. Jacek Wszołek
mgr inż. Marcin Wiśniewski
inż. Rafał Chrabąszcz
inż. Andrzej Kamisiński
inż. Jehoszafat Zimnowoda

**Politechnika Śląska:**
dr hab inż. Andrzej Chydziński (kierownik *Platformy Xen*)
dr inż. Agnieszka Brachman
mgr inż. Błażej Adamczyk
mgr inż. Łukasz Chróst
mgr inż. Tomasz Fortuna

# Spis treści

**Przedmowa** ..... III

**Wykonawcy projektu IIP** ..... IV

## System IIP

Infrastruktura bazowa dla Systemu IIP ..... 1
*Wojciech Burakowski, Halina Tarasiuk, Andrzej Bęben, Janusz Granat*

Implementacja wirtualizacji dla Systemu IIP na platformie EZappliance .. 11
*Piotr Wiśniewski, Damian Parniewicz, Krzysztof Dombek, Artur Juszczyk, Iwo Olszewski*

Realizacja węzła Systemu IIP z wykorzystaniem Xen ..... 25
*Błażej Adamczyk, Andrzej Chydziński*

Współpraca systemu zarządzania z modułem wymiarowania zasobów w Systemie IIP ..... 38
*Janusz Granat, Jakub Bielecki, Wojciech Szymak, Krzysztof Wajda, Mirosław Kantor, Janusz Gozdecki, Jacek Rak*

Implementacja i testy architektury bezpieczeństwa na poziomie 2 Systemu IIP ..... 46
*Jerzy Konorski, Krzysztof Cabaj, Zbigniew Kotulski, Paweł Szałachowski, Grzegorz Kołaczek, Jerzy Kasperek, Piotr Pacyna, Dominik Rzepka, Wojciech Romaszkan, Marcin Rupiński, Jehoszafat Zimnowoda, Andrzej Kamisiński, Paweł Rajda*

Specyfikacja i implementacja modułu wymiarowania zasobów w Systemie IIP ..... 77
*Krzysztof Wajda, Mirosław Kantor, Janusz Gozdecki, Jacek Rak*

## Równoległy Internet IPv6 QoS

Równoległy Internet IPv6 QoS: system sygnalizacji i scenariusze aplikacyjne ..... 87
*Halina Tarasiuk, Jan Rogowski, Paweł Świątek, Krzysztof Brzostowski, Jarosław Drapała, Krzysztof Juszczyszyn, Cezary Mazurek, Bartosz Lewandowski, Arkadiusz Radziuk, Marcin Wrzos, Dariusz Walczak, Marek Natkaniec, Janusz Gozdecki, Marek Sikora*

Implementacja płaszczyzny danych i płaszczyzny sterowania dla Równoległego Internetu IPv6 QoS . . . . . 105
*Piotr Wiśniewski, Halina Tarasiuk, Mariusz Rawski, Iwo Olszewski, Łukasz Ogrodowczyk, Sławomir Hanczewski, Adam Kaliszan, Agnieszka Brachman, Krzysztof Gierłowski*

Aplikacje testowe i ich odwzorowanie na klasy usług w Równoległym Internecie IPv6 QoS . . . . . 119
*Halina Tarasiuk, Wojciech Burakowski, Marek Natkaniec, Janusz Gozdecki, Krzysztof Łoziak, Katarzyna Kosek-Szott, Szymon Szott, Adam Grzech, Paweł Świątek, Bartosz Lewandowski, Cezary Mazurek, Arkadiusz Radziuk, Dariusz Walczak*

**Równoległy Internet Sieć Świadoma Przekazywanej Treści**

Sieci świadome przekazywanej treści . . . . . 135
*Andrzej Bęben*

Realizacja sieci świadomej treści w oparciu o wirtualne zasoby Systemu IIP 146
*Piotr Krawiec, Andrzej Bęben, Piotr Wiśniewski, Jordi Mongay Batalla, Jakub Borzdyński, Jakub Gutkowski, Piotr Ostapowicz, Andrzej Kozik, Radosław Rudek*

Routing i wymiarowanie w sieci świadomej treści . . . . . 165
*Paweł Białoń, Janusz Granat, Paweł Olender, Andrzej Beben, Piotr Krawiec, Piotr Wiśniewski, Bartosz Belter, Jakub Gutkowski, Łukasz Łopatowski*

Aplikacje Sieci Świadomej Treści . . . . . 179
*Michał Kosiedowski, Ewa Kuśmierek, Aleksander Stroiński, Michał Cholewa, Przemysław Głomb, Sebastian Opozda, Michał Romaszewski, Arkadiusz Sochan, Andrzej Bęben, Piotr Krawiec*

**Sieć badawcza PL-LAB**

Portal PL-LAB . . . . . 194
*Łukasz Dolata, Radosław Krzywania, Miłosz Przywecki, Piotr Krawiec*

**Transformacja z IPv4 do IPv6**

Scenariusze biznesowe w środowisku IPv6 . . . . . 209
*Jordi Mongay Batalla, Piotr Krawiec*

Zarządzanie mobilnością w nowoczesnych sieciach IP oraz sieciach Internetu Przyszłości . . . . . 220

*Józef Woźniak, Krzysztof Gierłowski, Tomasz Gierszewski, Michał Hoeft*

Dobre praktyki automatyzacji procesu migracji do protokołu IPv6 . . . . . 262
*Krzysztof Nowicki, Wojciech Gumiński, Paweł Kamiński, Łukasz Łoska, Karol Podolski, Krzysztof Wasilewski*

Protokoły SNMP oraz IPsec w sieciach wykorzystujących IPv6 . . . . . 279
*Bartosz Gajda, Wiktor Procyk, Piotr Skałecki, Jan Kwiatkowski, Krzysztof Chudzik, Kamil Nowak*

(Nie)docenione rozwiązanie – routing za pomocą protokołu DHCPv6 w środowisku IPv6 . . . . . 293
*Krzysztof Nowicki, Mateusz Ożga, Tomasz Mrugalski*

Zasoby i eksperymenty w sieci PL-LABv6 . . . . . 308
*Józef Woźniak, Krzysztof Nowicki, Tomasz Mrugalski, Wojciech Gumiński, Henryk Gut, Jordi Mongay Batalla, Waldemar Latoszek, Artur Binczewski, Bartosz Gajda, Wiktor Procyk, Piotr Skałecki, Krzysztof Chudzik, Jan Kwiatkowski, Kamil Nowak*

**Wykaz autorów** . . . . . 317

# Infrastruktura bazowa dla Systemu IIP

Wojciech Burakowski[1], Halina Tarasiuk[1], Andrzej Bęben[1], Janusz Granat[2]

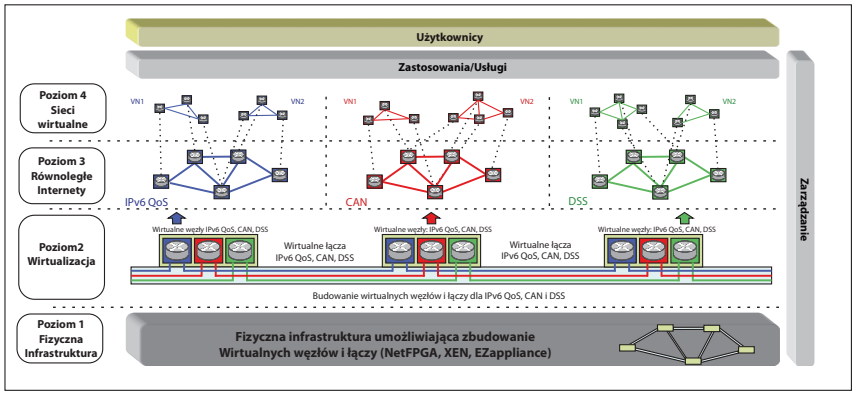
[1] Instytut Telekomunikacji, Politechnika Warszawska
[2] Instytut Łączności

**Streszczenie** W artykule przedstawiono infrastrukturę bazową dla Systemu IIP, której zadaniem jest stworzenie warunków technicznych dla ustanowienia tzw. Równoległych Internetów. Infrastruktura ta obejmuje zarówno sieć szkieletową, jak i sieci dostępowe. W sieci szkieletowej rolę węzłów sieci stanowią urządzenia umożliwiające wirtualizację, które są połączone systemem transmisyjnym umożliwiającym wydzielenie zasobów dla poszczególnych Równoległych Internetów. Ponadto, w artykule pokrótce opisano podstawowe funkcjonalności omawianej infrastruktury, które dotyczą takich aspektów jak zarządzanie, wymiarowanie i zapewnienie bezpieczeństwa.

## 1 Wprowadzenie

Artykuł ten opisuje infrastrukturę bazową Systemu IIP, która została zaprojektowana i zaimplementowana w ramach projektu *„Inżynieria Internetu Przyszłości”*. System IIP jest propozycją realizacji nowej infrastruktury sieciowej dla Internetu, która zakłada wykorzystanie wirtualizacji jako techniki umożliwiającej uzyskanie nowych własności sieci, nieosiągalnych przy obecnej technice TCP/IP. W szczególności, System IIP pozwala na zbudowanie w sieci wielu Równoległych Internetów, które mogą działać w oparciu o inne protokoły komunikacyjne. Implementowanymi Równoległymi Internetami są: (1) IPv6 QoS, który zapewnia jakość przekazu pakietów dla wybranych strumieni ruchu, co uzyskuje się dzięki zaimplementowaniu tzw. klas usług wraz z koniecznymi mechanizmami QoS, (2) CAN *(ang. Content Aware Networks)* dla efektywnego przekazu treści w sieci, co uzyskujemy poprzez rozważanie wiedzy o miejscach, gdzie dana treść jest dostępna oraz wiedzy o warunkach ruchowych panujących w sieci, (3) DSS *(ang. Data Streams Switching)* dla przekazu strumieni ruchu wymagających dużych przepływności i wysokiej jakości przekazu danych. Przyjęta architektura systemu jest przedstawiona na Rys. 1 i obejmuje 6 warstw. Dokładny opis architektury został przedstawiony m.in. w [1], [2] i [3].

Artykuł skupia się na zagadnieniach związanych z realizacją infrastruktury bazowej, która daje możliwość ustanowienia pewnej liczby Równoległych Internetów. Internety te muszą działać w izolacji i bazować na łączach i węzłach wirtualnych. Poziom 1/2 architektury, który będzie w tym artykule omawiany, ma za zadanie ustanowienie łączy wirtualnych o zadanej przepływności dla każdego Równoległego Internetu oraz stworzenie środowiska dla ustanowienia węzłów wirtualnych.

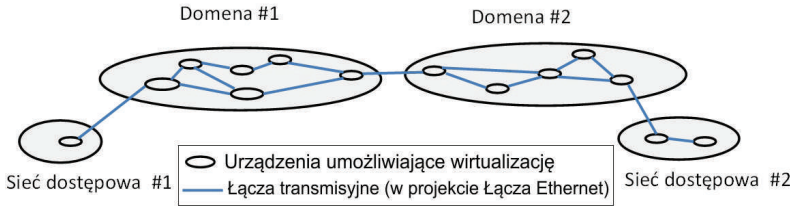



**Rysunek 1.** Architektura Systemu IIP

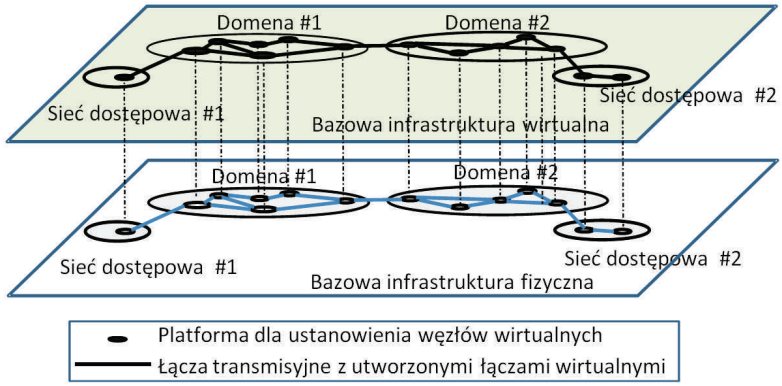
Artykuł jest zorganizowany w następujący sposób. W Rozdziale 2. opisane zostały pokrótce podstawy bazowej infrastruktury wirtualnej Systemu IIP. Elementy bazowej infrastruktury wirtualnej takie jak sieć szkieletowa, podsystem zarządzania infrastrukturą bazową oraz elementy podsystemu bezpieczeństwa są opisane w Rozdziale 3. Rozdział 4. omawia zagadnienia związane ze stworzeniem środowiska wirtualnego pozwalającego na ustanowienie łączy wirtualnych i platformy dla ustanowienia węzłów wirtualnych. Ostatecznie, Rozdział 5. podsumowuje artykuł.

## 2 Podstawy bazowej infrastruktury wirtualnej

Ogólny schemat infrastruktury bazowej Systemu IIP przedstawia Rys. 2. Na rysunku tym zaznaczono przykładowo dwie domeny stanowiące sieć szkieletową, przy czym w każdej z nich węzłami sieciowymi są urządzenia umożliwiające wirtualizację, takie jak EZappliance, NetFPGA lub XEN. Urządzenia te są połączone łączami transmisyjnymi, w projekcie są to łącza Ethernet. Do domen są




**Rysunek 2.** Ogólny schemat infrastruktury fizycznej Systemu IIP



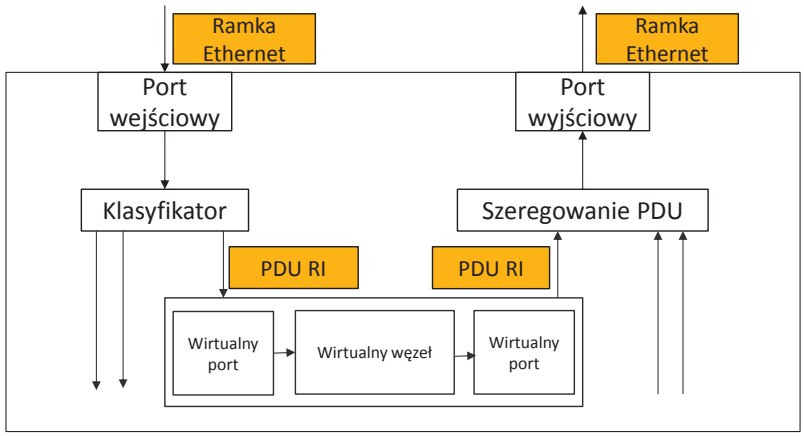

**Rysunek 3.** Tworzenie infrastruktury wirtualnej na bazie infrastruktury fizycznej

przyłączone sieci dostępowe, a do sieci dostępowych są przyłączone urządzenia końcowe.

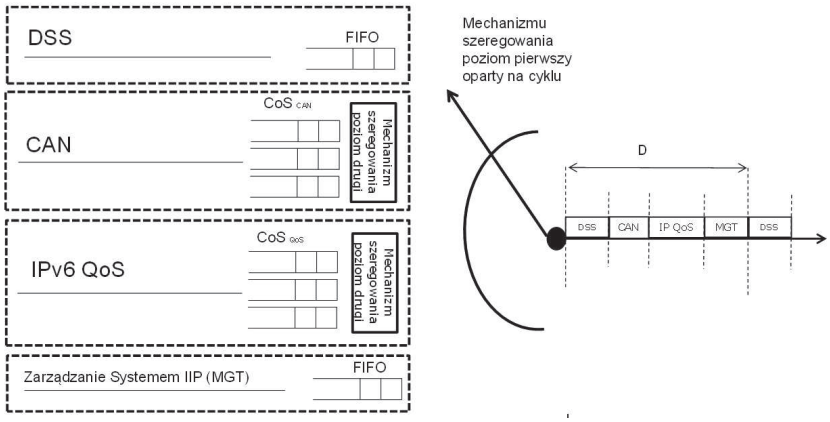
Wirtualizacja infrastruktury fizycznej polega na utworzeniu łączy wirtualnych na każdym łączu transmisyjnym oraz na stworzeniu platformy do ustanowienia węzłów wirtualnych, tak jak to pokazano na Rys. 3. Stworzenie platformy dla ustanowienia węzłów wirtualnych w urządzeniach umożliwiających wirtualizację wymagało zaprojektowania wielu mechanizmów, które są wyszczególnione na Rys. 4. Przychodzące bloki do danego portu wejściowego, które stanowią pole danych w ramkach Ethernet, są kierowane do klasyfikatora, który przekazuje je do danego węzła wirtualnego w zależności od wartość nagłówka (nagłówek Równoległego Internetu - RI). Po przetworzeniu bloku PDU *(ang. Packet Data Unit)* przez węzeł wirtualny blok PDU RI jest przekazywany do mechanizmu szeregowania bloków.

Dla utworzenia łączy wirtualnych na łączach fizycznych stosuje się specjalne szeregowania bloków PDU [4], przesyłanych w ramach poszczególnych Równoległych Internetów. Szeregowanie tych bloków odbywa się w ten sposób, iż na łącze fizyczne nakładamy multipleksację statyczną z podziałem czasu. Wprowadza się cykl czasowy o zadanej długości, w którym rozróżniamy okresy czasowe dla przekazu bloków przesyłanych w ramach danego Równoległego Internetu lub podsystemu zarządzania. W efekcie, nawet jeżeli dany okres czasowy nie jest wykorzystywany to nie może być on użyty dla przekazu innych bloków niż te dla którego jest on dedykowany. Schemat działania szeregowania bloków na poziomie pierwszym przedstawia Rys. 5. Drugi poziom szeregowania bloków dotyczy Równoległych Internetów i jest realizowany w węzłach wirtualnych.

Rys. 6 oraz Rys. 7 przedstawiają odpowiednio algorytmy realizacji mechanizmów szeregowania dla poziomu pierwszego i drugiego. Algorytmy te zostały zaimplementowane na platformach NetFPGA i XEN. Na platformie EZappliance działanie proponowanego mechanizmu szeregowania PDU dla poziomu pierw-




**Rysunek 4.** Schemat blokowy urządzenia umożliwiającego wirtualizację dla ustanowienia węzłów wirtualnych

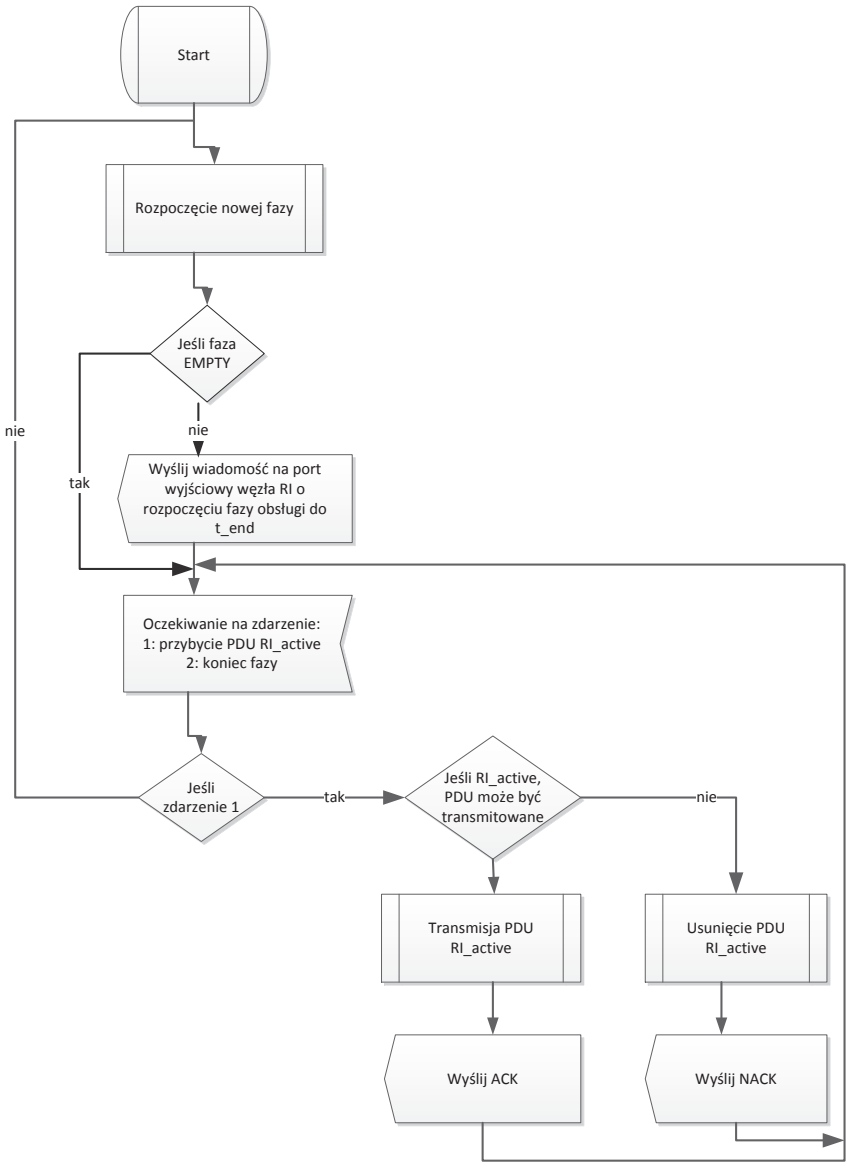



**Rysunek 5.** Tworzenie łączy wirtualnych na łączach fizycznych, $D$ - długość cyklu, CoS *(ang. Classes of Service)* - klasy usług

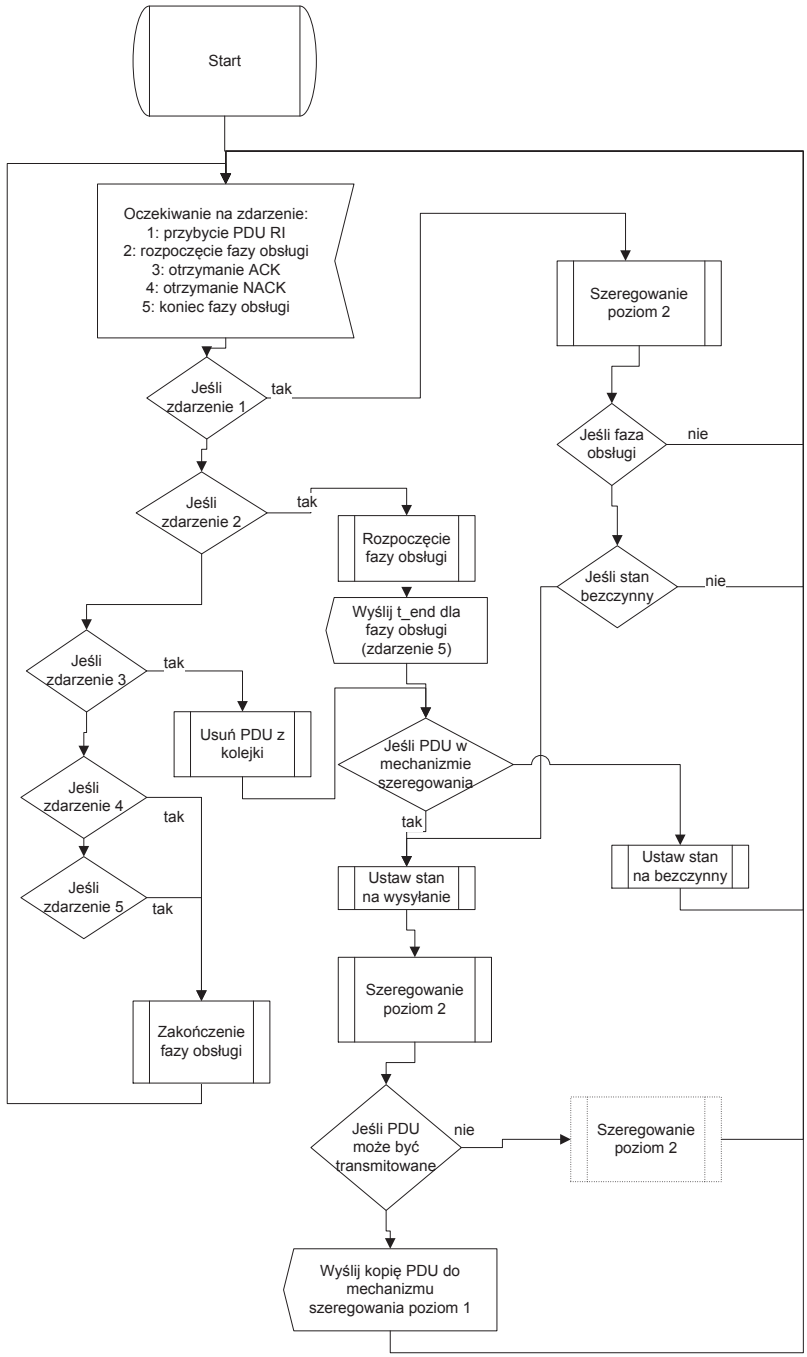
szego emulujemy poprzez odpowiednią konfigurację mechanizmu DRR *(ang. Deficit Roud Robin)* i kształtowania ruchu, gdyż nie ma możliwości realizacji innych niż wbudowane mechanizmów szeregowania PDU.

Na potrzeby implementacji algorytmów przyjęto następujące oznaczenia: EMPTY - faza pusta - faza w cyklu, w której nie są generowane żadne PDU; RI_active - faza aktywna (obsługi) dla danego RI: RI_active=IPv6 QoS, CAN, DSS; t_end - czas zakończenia fazy aktywnej, ACK - potwierdzenie wysłania

PDU, NACK - potwierdzenie negatywne - oznacza brak możliwości wysłania PDU.




**Rysunek 6.** Algorytm szeregowania PDU dla poziomu pierwszego




**Rysunek 7.** Algorytm szeregowania PDU dla poziomu 2.

# 3 Elementy bazowej infrastruktury wirtualnej

W rozdziale tym opiszemy pokrótce rozważane w projekcie elementy bazowej infrastruktury wirtualnej. Infrastruktura ta składa się z sieci szkieletowej (wielodomenowej) i sieci dostępowych. Obie te sieci powinny być infrastrukturami wirtualnymi, tak jak to przedstawiono w rozdziale 2. O ile w przypadku sieci szkieletowej jest to w pewnym sensie zadanie łatwiejsze, co wynika bezpośrednio z faktu, iż w tym przypadku, stosujemy nowe platformy umożliwiające wirtualizację, to w przypadku sieci dostępowych stosujemy „dodatkowe" rozwiązania dla już eksploatowanego sprzętu sieciowego [16], [8], [9]. Dlatego też, w przypadku sieci dostępowych nie możemy w chwili obecnej oczekiwać zadowalających rozwiązań z punktu widzenia wymagań stawianych Internetowi Przyszłości. Ponadto, oprócz wymienionych sieci, w projekcie skonstruowano rozwiązania dla podsystemu zarządzania i podsystemu bezpieczeństwa

## 3.1 Sieć szkieletowa

Rozwiązania dla wirtualizacji sieci szkieletowej opierają się w zasadzie na zastosowaniu nowych jakościowo urządzeń, tj. urządzeń umożliwiających wirtualizację węzłów i łączy. Jak wspomniano powyżej, tymi urządzeniami są: (1) EZappliance, który jest zaawansowanym urządzeniem i ma już wiele wbudowanych mechanizmów, jak choćby mechanizmy szeregowania bloków [5], (2) NetFPGA, który należało zaprojektować „od podstaw", przy czym wiele złożonych funkcji należało oprogramować, natomiast rozwiązania sprzętowe realizują jedynie funkcje podstawowe [6], (3) Xen, czyli programistyczna platforma otwarta, która pozwala na oprogramowanie całego rozwiązania dla Systemu IIP, natomiast należy się liczyć ze stosunkowo małą wydajnością [7].

## 3.2 Podsystem zarządzania

Jednym z kluczowych elementów nowych architektur Internetu Przyszłości są moduły zarządzania. W realizowanych obecnie projektach badawczych [10], [11] jak również w prognozach przyszłych badań [3] podkreśla się znaczenie autonomii w zarządzaniu w zadaniach konfiguracji, optymalizacji wykorzystania zasobów, reagowania na uszkodzenia i ataki zewnętrzne *(self-configuring, self-optimizing self-healing, self-protecting)*.

Architektura zarządzania Systemu IIP została zaprojektowana tak, aby w przyszłości umożliwić realizację zaawansowanych algorytmów. W pierwszym kroku skupiono jednak uwagę na funkcjonalności podstawowej. Tego typu podejście pozwoliło na szybkie testowanie nowo tworzonej architektury Systemu IIP. W ramach funkcjonalności podstawowej system zarządzania dla Systemu IIP wspomaga następujące procesy: (1) obsługę żądań utworzenia Równoległych Internetów, (2) przydział zasobów na podstawie przyjętego mechanizmu wymiarowania, (3) monitorowanie wykorzystania zasobów, oraz (4) zwolnienie zasobów Równoległego Internetu.

Moduł obsługi żądań utworzenia Równoległego Internetu ma za zadanie przyjęcie żądania przydzielenia zasobów od operatora Równoległego Internetu PIO *(ang. Parallel Internet Operator)*. Operator w żądaniu określa lokalizację węzłów brzegowych, ich identyfikatory, oraz liczbę interfejsów dla sieci dostępowych. W Równoległym Internecie CAN dodatkowo określa się również lokalizację węzłów szkieletowych. Następnie określa się wymagane przepustowości miedzy węzłami brzegowymi lub przepustowości łączy w przypadku definiowania pełnej topologii Równoległego Internetu.

Po przyjęciu żądań utworzenia Równoległych Internetów są uruchamiane algorytmy wymiarowania. Zrealizowano dwa moduły wymiarowania. Moduł pierwszy oblicza przydział zasobów zgodnie ze zgłoszonymi żądaniami utworzenia Równoległych Internetów. Moduł ten dokonuje wyboru lokalizacji węzłów oraz przydziela przepustowości dla poszczególnych łączy. W chwili obecnej dostępne są algorytmy bazujące na odpowiednio sformułowanych zadaniach optymalizacji. Zadania optymalizacji [12] wykorzystują kryteria minimalizacji zajętej przepustowości całkowitej (z opcją minimalizacji opóźnienia transmisyjnego) oraz minimalizacji użycia zasobów węzłowych. Do obliczeń wykorzystywane są dane o zasobach zgromadzone w bazie danych zarządzania. Zadaniem modułu drugiego jest obliczenie parametrów specyficznych dla poszczególnych platform wirtualizacyjnych, które są wykorzystywane przez narzędzia konfiguracji. Integracja modułów zarządzania i wymiarowania została opisana w pracy [13].

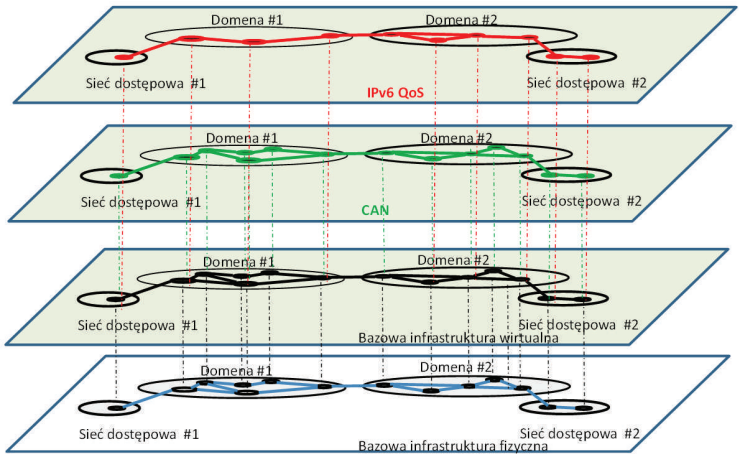
Do konfigurowania platform wirtualizacyjnych wykorzystywany jest CLI *(ang. Command Line Interface)*. Działanie CLI jest możliwe dzięki opracowaniu dedykowanych adapterów zarządzania dla każdej z platform wirtualizacyjnych. Dodatkowo zrealizowano funkcjonalność monitorowania wykorzystania zasobów oraz zwolnienia zasobów Równoległego Internetu.

### 3.3 Podsystem bezpieczeństwa

Podsystem bezpieczeństwa Systemu IIP jest oparty na trzech poziomach zabezpieczeń [14]. Poziom 1 obejmuje sprzętową ochronę systemu transmisyjnego w sieci szkieletowej Systemu IIP. Poziom 2 architektury bezpieczeństwa obejmuje analizę anomalii wykrytych na poziomie 1, anomalii w sieci zarządzania systemu oraz nieoczekiwanego zachowania węzła. Dla każdego węzła jego moduł LAD *(ang. Local Anomaly Detection)* oblicza metryki reputacyjne, które są przekazywane do poziomu 3. Poziom 3 architektury bezpieczeństwa analizuje nadesłane metryki z poszczególnych węzłów Systemu IIP w kontekście bezpieczeństwa całego systemu. Poziom 1 architektury bezpieczeństwa został zrealizowany w postaci wydzielonych modułów przeznaczonych do ochrony warstwy łącza danych pomiędzy węzłami sieci szkieletowej [15]. Pakiet opuszczający węzeł IIP jest przetwarzany przez sprzętowy koder, który dodaje do przekazywanej ramki sygnaturę skrótu HMAC-SHA-1 wraz z numerem sekwencyjnym pakietu. Po stronie odbiorczej przychodzące pakiety są weryfikowane pod względem poprawności sygnatury i numeru sekwencyjnego, stwierdzone nieprawidłowości są raportowane do poziomu 2 architektury bezpieczeństwa, zaś prawidłowe pakiety zostają przekazane do odbiornika węzła IIP.

## 4 Implementacja Równoległych Internetów

Sposób ustanowienia Równoległych Internetów na wirtualnej infrastrukturze bazowej przedstawia Rys. 8. Każdy z Równoległych Internetów działa w oparciu o łącza i węzły wirtualne. Dla ustanowienia danego Równoległego Internetu korzystamy w sposób bezpośredni z bazowej infrastruktury wirtualnej. Na Rys. 8. przedstawiono przykład sieci CAN i IPv6 QoS. Sieci te mogą mieć różną topologię, w zależności od przydzielonych tym sieciom zasobów.




**Rysunek 8.** Ustanowienie Równoległych Internetów CAN i IPv6 QoS na wirtualnej infrastrukturze bazowej

## 5 Podsumowanie

W artykule przedstawiono pokrótce bazową infrastrukturę, którą założyliśmy w Systemie IIP. Główne elementy tej infrastruktury to sieć szkieletowa, która wymagała opracowania nowych elementów sieciowych bazujących na urządzeniach umożliwiających wirtualizację. Głównym rozwiązaniem umożliwiającym wirtualizację łączy jest zastosowanie dwupoziomowego mechanizmu szeregowania bloków danych, który został opisany w artykule.

## Literatura


1. W.Burakowski, H.Tarasiuk, A.Bęben, *Architektura Systemu IIP*, artykuł plenarny, 27 Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź, 2011
2. W. Burakowski, H. Tarasiuk, A. Beben, G. Danilewicz, *Virtualized network infrastructure supporting co-existence of Parallel Internets*, In Proc. of SNPD 2012, Kyoto, August 2012, pp. 679-684.
3. W.Burakowski, i inni, *The Future Internet Engineering project in Poland: goals and achievements*, Towards and Beyond Europe 2020 - the Significance of Future Internet for Regional Development, Future Internet Week Report, Publishing House of the Institute for Sustainable Technologies - National Research Institute, 2012
4. W. Burakowski, i inni, *Idealne urządzenie umożliwiające wirtualizację infrastruktury sieciowej w Systemie IIP*, KSTiT, Łódź, 2011
5. P. Wiśniewski i inni, *Implementacja wirtualizacji dla Systemu IIP na platformie EZappliance*, KSTiT, Warszawa 12-14 września 2012
6. M. Rawski, P. Szotkowski, P. Tomaszewicz, *Sprzętowa realizacja platformy wirtualizacji dla Systemu IIP wykorzystująca NetFPGA*, KSTiT, Warszawa 12-14 września 2012
7. B. Adamczyk, A. Chydziński, *Implementacja programowej platformy wirtualizacji dla Systemu IIP z wykorzystaniem monitora maszyn wirtualnych*, KSTiT, Warszawa 12-14 września 2012
8. M. Rodziewicz, K. Bąkowski, R. Krenz, *Wieloskokowa bezprzewodowa siec dostępowa w projekcie Inżynieria Internetu Przyszłości*, KSTiT, Warszawa 12-14 września 2012
9. P. Szczepański, R. Piramidowicz, K. Welikow, M. Bugaj, P. Zwierko, *System dostępowy GEPON na potrzeby Internetu Przyszłości*, KSTiT, Warszawa 12-14 września 2012
10. J. O. Kephart, D. M. Chess, *The vision of autonomic computing*, Computer, vol. 36, no. 1, pp. 41-50, 2003.
11. Z. Movahedi, M. Ayari, R. Langar,G. Pujolle, A Survey of Autonomic Network Architectures and Evaluation Criteria, Communications Surveys & Tutorials, IEEE, vol.14, no.2, pp.464-490, Second Quarter 2012
12. J. Gozdecki, M. Kantor, K. Wajda, J. Rak, *A Flexible Provisioning Module for Optimizing Utilization of Resources for the Future Internet IIP Initiative*, Networks 2012, Rzym 2012
13. J. Granat, i inni, *Integracja systemu zarządzania z modułem wymiarowania zasobów w Systemie IIP*, KSTiT 2012
14. K. Cabaj, Z. Kotulski, P. Szałachowski, G. Kołaczek, J. Konorski, *Implementation and testing of Level 2 security architecture for the IIP System*, KSTiT, Warszawa 12-14 września 2012
15. J. Konorski, i inni, *Implementacja sprzętowa modułu HMAC-SHA-1 do ochrony komunikacji w systemie IIP*, Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Warszawa 12-14 września 2012
16. R. Krenz, i inni, *Koncepcja systemów dostępowych do Internetu Przyszłości*, Inżynieria Internetu Przyszłości część I, Praca pod redakcją W. Burakowskiego i P. Krawca, Warszawa 2012

# Implementacja wirtualizacji dla Systemu IIP na platformie EZappliance

Piotr Wiśniewski[1], Damian Parniewicz[2], Krzysztof Dombek[2], Artur Juszczyk[2], Iwo Olszewski[2]

[1] Instytut Telekomunikacji, Politechnika Warszawska
[2] Poznańskie Centrum Superkomputerowo-Sieciowe

**Streszczenie** Artykuł przedstawia projekt i implementację warstwy wirtualizacji na programowalnej platformie sprzętowej EZappliance opartej na procesorze sieciowym EZchip NP-3. Warstwa wirtualizacji umożliwia niezależną koegzystencję kilku węzłów wirtualnych należących do technologicznie zróżnicowanych sieci (zwanych Równoległymi Internetami) na jednym urządzeniu fizycznym. Na chwilę obecną na platformie EZappliance zostały wdrożone węzły wirtualne należące do Równoległego Internetu IPv6 QoS (sieć z gwarancją jakości usług) oraz CAN (sieć świadoma przekazywanej treści).

## 1 O platformie EZappliance

EZappliance jest urządzeniem produkowanym przez firmę EZchip Technologies [5] dedykowanym do szybkiego tworzenia i wdrażania wydajnych aplikacji sieciowych [4]. Jest to kompaktowa platforma sprzętowa składająca się z programowalnego procesora sieciowego EZchip NP-3 o sumarycznej przepustowości 30Gbps oraz z wbudowanego systemu Host CPU. Procesor NP-3 jest centralnym elementem tej platformy i umożliwia realizację parsowania, klasyfikacji, modyfikacji, przełączania i zarządzania przetwarzanych pakietów [4]. Procesor EZchip NP-3 jest podłączony za pomocą interfejsu PCI do wbudowanego systemu (Host CPU) opartego o procesor ogólnego przeznaczenia PowerQUICK™ III z systemem Linux (dystrybucja ELDK v4.2). Dodatkowe oprogramowanie zainstalowane w obrębie wbudowanego systemu Linux pozwala na zarządzanie i kontrolowanie pracy przetwarzania ramek w procesorze sieciowym NP-3. Urządzenia EZappliance wykorzystywane w projekcie IIP są wyposażone w 24 porty Gigabit Ethernet i dodatkowo w razie potrzeby mogą być wyposażone w dwa interfejsy 10-Gigabit Ethernet (zobacz Rys. 1).

Urządzenia EZappliance służą do tworzenia aplikacji takich jak punkty dostępu do sieci Carrier Ethernet, routery i przełączniki sieci metropolitalnych, bramy bezpieczeństwa, urządzenia monitorujące i analizujące ruch sieciowy. Producent udostępnia konkretne przykłady oprogramowania na EZappliance zawierającego funkcjonalności związane z następującymi technologiami: przełączanie L2,
Q-in-Q ala (ang. Queue in Queue), PBT (ang. Provider Backbone Transport),

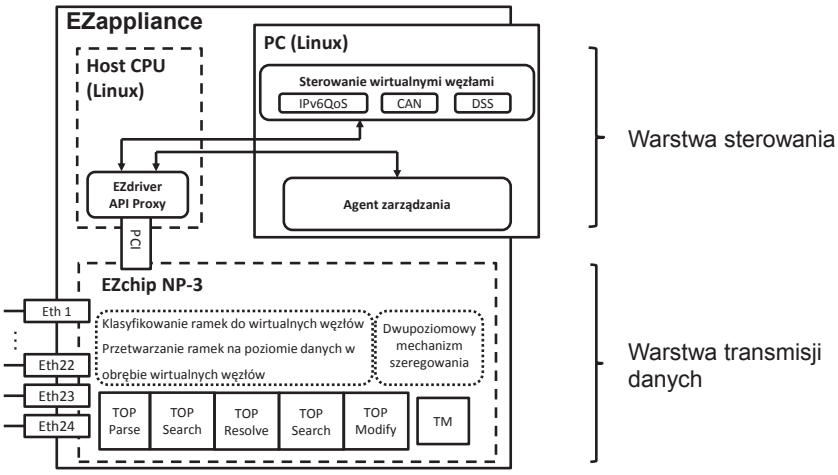
**Rysunek 1.** Urządzenie EZappliance NP-3 wykorzystywane w Systemie IIP.

MPLS (ang. Multiprotocol Label Switching), T-MPLS (ang. Transport MPLS), VPLS (ang. Virtual Private Local Area Service), ruting IPv4/IPv6 i IDS (ang. Intrusion Detection System). W dalszej części artykułu przedstawiono w jaki sposób urządzenie EZappliance zostało wykorzystane w projekcie IIP. W szczególności, rozdział drugi opisuje realizację węzła Systemu IIP z wykorzystaniem dostępnych możliwości programowych i konfiguracyjnych platformy EZappliance. Kolejne dwa rozdziały przedstawiają szczegóły implementacji wirtualizacji warstwy danych oraz wirtualizacji warstwy sterowania.

## 2 Realizacja węzła Systemu IIP na urządzeniu EZappliance

System IIP został opracowany w ramach projektu Inżynieria Internetu Przyszłości (IIP) [1,2]. System został zrealizowany w oparciu o architekturę warstwową zaproponowaną w [3]. Dwie pierwsze warstwy architektury (warstwa fizyczna i wirtualizacji) mają na celu dostarczenie infrastruktury wirtualnej dla Równoległych Internetów - RI (ang. Parallel Internets). Aktualnie na platformie EZappliance zostały zaimplementowane wirtualne węzły dwóch typów Równoległych Internetów: węzły brzegowe[3] IPv6 QoS (Quality of Service), czyli sieci z gwarancją jakości przekazu pakietów, oraz węzły CAN (Content Aware Network), czyli sieci świadomej przekazywanej treści. W najbliższych planach jest uruchomienie także trzeciego typu Równoległego Internetu: DSS (Data Stream Switching) realizującego komutację strumieni danych. Powyższe trzy Równoległe Internety różnią się zarówno warstwą protokołów dla przekazu danych, jak i warstwą protokołów sterowania. Implementacja węzła Systemu IIP (patrz Rys. 2) wykorzystuje

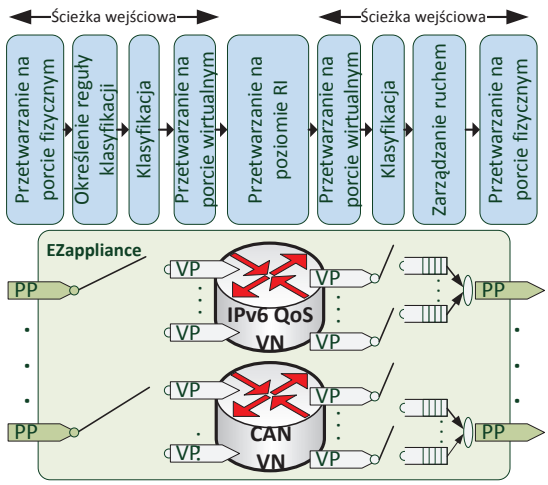
[3] Oprócz węzłów brzegowych w Systemie IIP implementowane są węzły szkieletowe w oparciu o platformę sprzętową NetFPGA i platformę czysto programową opartą o oprogramowanie XEN.




**Rysunek 2.** Schemat realizacji węzła systemu IIP na platformie EZappliance.

zarówno procesor sieciowy NP-3 jak i wbudowany system Linux. Dodatkowo, ze względu na małą ilość dostępnej pamięci na platformie EZappliance został wykorzystany zewnętrzny serwer PC, gdzie umieszczono oprogramowane warstwy sterowania dla węzłów RI oraz oprogramowanie agenta zarządzania warstwą wirtualizacji.

Przy implementacji warstwy transmisji danych w obrębie węzła Systemu IIP, do przetwarzania datagramów wykorzystano wszystkie rodzaje wyspecjalizowanych, programowalnych procesorów zoptymalizowanych zadaniowo (TOP) umieszczonych w układzie scalonym NP-3: TOPparse, TOPsearchI, TOPresolve, TOPsearchII i TOPmodify. Procesory te realizują zarówno przetwarzanie jednostek danych PDU (ang. Protocl Data Unit) na poziomie wirtualizacji węzła jak i przetwarzanie jednostek PDU w obrębie poszczególnych wirtualnych węzłów. Dodatkowo wykorzystane zostały mechanizmy modułu TM (Traffic Manager), również ulokowanego w obrębie procesora sieciowego NP-3. Mechanizmy te pozwalają na sprzętowe zarządzanie ruchem poprzez: pomiar, znakowanie i szeregowanie pakietów oraz kształtownie profilu ruchowego strumienie pakietów. Konfiguracja modułu TM, wczytywanie programów dla poszczególnych procesorów TOP i zarządzanie parametrami pracy tych programów odbywa się za pomocą oprogramowania EZdriver API zainstalowanego na wbudowanym systemie Linux. Interfejs EZdriver API jest wykorzystywany przez oprogramowanie warstwy sterowania. Warstwa sterowania składa się z agenta zarządzania, który zarządza zasobami fizycznymi i wirtualizacją tych zasobów, oraz oprogramowania sterującego węzłami wirtualnymi każdego z Równoległych Internetów: IPv6 QoS, CAN i DSS.




**Rysunek 3.** Obsługa ramek na platformie EZappliance.

## 3 Wirtualizacja płaszczyzny danych

Warstwa wirtualizacji na urządzeniu EZappliance umożliwia jednoczesne i niezależne działanie węzłów Równoległych Internetów. Warstwa wirtualizacji dostarcza interfejs w postaci wirtualnych portów oraz kolejek (oraz stowarzyszonych rejestrów) za pomocą których RI-ty mogą przesyłać swoje jednostki danych. Dzięki temu interfejsowi protokoły oraz mechanizmy wykorzystywane w warstwach: fizycznej oraz wirtualizacji są całkowicie niezależne od warstwy Równoległych Internetów. Logika działania każdego RI dla każdego procesora zadaniowego TOP znajduje się w niezależnym pliku z kodem asemblerowym. Warstwa wirtualizacji na każdym procesorze TOP wywołuje we właściwym momencie kod odpowiedniego RI. Obsługa ramek na urządzeniu EZappliance składa się trzech głównych logicznych etapów, jak to przedstawiono na Rys. 3:

- przetwarzanie ramek na poziomach fizycznym i wirtualizacji - ścieżka wejściowa,
- przetwarzanie ramek na poziomie Równoległych Internetów,
- przetwarzanie ramek na poziomach fizycznym i wirtualizacji - ścieżka wyjściowa.

### 3.1 Przetwarzanie ramek na poziomach fizycznym i wirtualizacji - ścieżka wejściowa

Obsługa ramek na ścieżce wejściowej składa z czterech etapów: przetwarzania na porcie fizycznym, określeniu reguły klasyfikacji, klasyfikacji oraz przetwarzania na porcie wirtualnym. Przetwarzanie na porcie fizycznym polega na odbiorze ramki Ethernet-owej z medium transmisyjnego (światłowodu lub kabla miedzianego), podobnie jak dla klasycznych urządzeń sieciowych. Określenie reguły klasyfikacji ma na celu wyznaczenie na podstawie jakiej reguły odebrana na pocie fizycznym ramka ma być przełączona do odpowiedniego portu wirtualnego danego Równoległego Internetu. Dopuszczalne są następujące reguły klasyfikacji (dla każdego portu fizycznego musi zostać predefiniowana reguła klasyfikacji):

- na podstawie nagłówka RI oraz identyfikatora portu fizycznego (reguła wykorzystywana na portach w kierunku sieci szkieletowej);
- na podstawie identyfikatora portu fizycznego (reguła wykorzystywana na portach w kierunku sieci dostępowej), wszystkie ramki odebranego na danym porcie fizycznym są wówczas przełączane do jednego RI;
- na podstawie pola VLan Tag (nagłówek 802.1Q) oraz identyfikatora portu fizycznego (reguła wykorzystywana na portach w kierunku sieci dostępowej), ramki przełączane do różnych RI mogą być odbierane na tym samym porcie fizycznym.

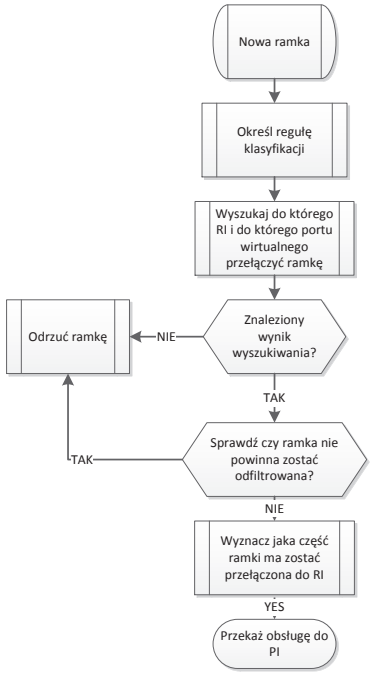
Trzecim etapem przetwarzania ramek jest klasyfikacja, która polega na przełączaniu ramki z portu fizycznego do portu wirtualnego zgodnie z wcześniej wyznaczoną regułą klasyfikacji. Ostatnim etapem obsługi ramki na ścieżce wejściowej jest przetwarzanie ramki na wirtualnym porcie wejściowym. Dla każdego portu wirtualnego określone jest jaka część ramki ma zostać przełączona do Równoległego Internetu (cała ramka, ramka bez nagłówka Ethernet-owego, ramka bez nagłówka Ethernet-owego i nagłówka PI) oraz czy przychodzące ramki mają być filtrowane. Możliwe jest filtrowanie ramek po adresach Ethernet-owych oraz po polu ethertype (pole nagłówka Ethernet-owego 802.3). W konsekwencji porty wirtualne oferują następującą funkcjonalność:

- usunięcie nagłówka Ethernet-owego,
- usunięcie nagłówka RI,
- odrzucenie ramek o niezgodnym typie (niezgodne pole ehertype),
- odrzucenie ramek o niezgodnym adresie źródłowym lub docelowym Ethernet.

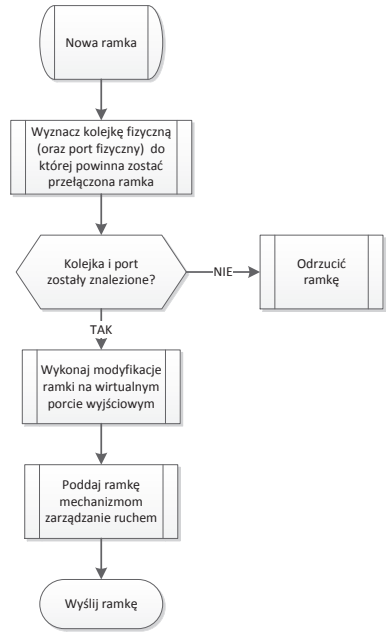
Całe przetwarzanie ramek na ścieżce wejściowej na poziomie fizycznym oraz poziomie wirtualizacji zostało zrealizowane na procesorze TOPparse. Uproszczony schemat obsługi ramki na procesorze TOPparse jest przedstawiony na Rys. 4.

### 3.2 Przetwarzanie ramek na poziomie Równoległych Internetów - przykład: Równoległy Internet IPv6 QoS

Węzeł Równoległego Internetu IPv6 QoS zaimplementowany na platformie EZappliance pełni funkcje routera brzegowego. Jest to specjalizowany router IPv6




Rys. 4a. Ścieżka wejściowa




Rys. 4b. Ścieżka wyjściowa

**Rysunek 4.** Uproszczony schemat przetwarzania ramki na ścieżce wejściowej oraz ścieżce wyjściowej.

z rozbudowanymi funkcjonalnościami kształtowania ruchu. Główne zadania węzła to klasyfikacja pakietów do odpowiedniej sieci wirtualnej, routing w ramach sieci wirtualnej, routing w ramach sieci szkieletowej, dodawanie i usuwanie nagłówka tunelu oraz kształtowanie ruchu. Węzeł Równoległego Internetu IPv6 QoS wymaga od warstwy L1/2 dostarczenie informacji o rodzaju portu (VPTP), przez który dany pakiet został odebrany. Wykorzystywane są 4 rodzaje portów: port do sieci dostępowej, port do sieci szkieletowej, wirtualny port do komunikacji z Host CPU oraz wirtualny port pętli zwrotnej. Dwa pierwsze to są portami danych, jednak rozróżnienie między nimi jest wymagane ze względu na różnice w obsłudze sieci szkieletowej i dostępowej w obrębie modułu routera IPv6. Wirtualny port do komunikacji z Host CPU wykorzystywany jest do przesyłania pakietów protokołów warstwy sterowania, czyli aktualnie protokołu OSPFv3. Ostatni z wymienionych portów jest bezpośrednio związany z kształtowaniem ruchu w RI IPv6 QoS. Część klas ruchu jest kształtowana per połączenie, co ze

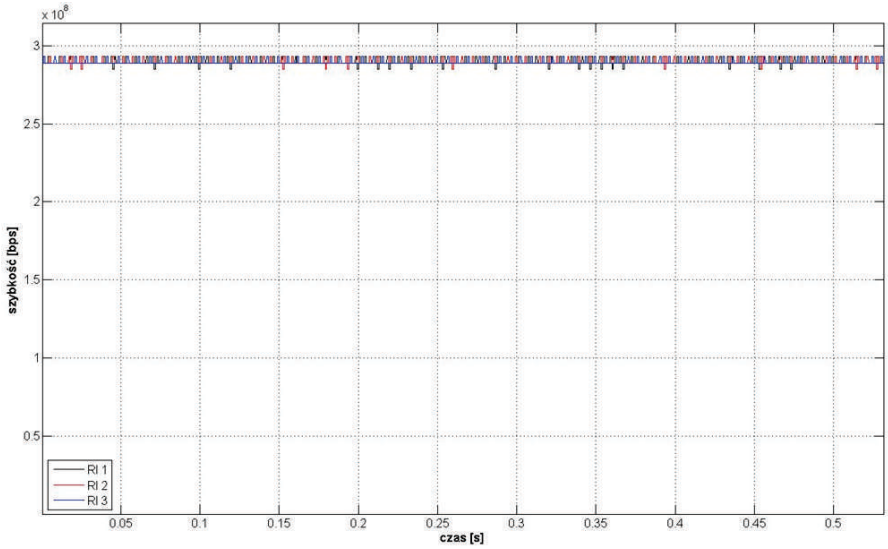
względu na ograniczenia TM wymaga podwójnego przejścia przez ten element. Po pierwszym przejściu pakiety są wysyłane na wirtualny port wejściowy i ponownie przetwarzane przez NP-3. Kolejną ważną z punktu widzenia warstwy L1/2 cechą węzła RI IPv6 QoS jest tunelowanie ruchu. Węzeł brzegowy dokleja i usuwa nagłówki tunelowe ("IP in IP") do pakietów danych, w związku z tym podczas przetwarzania tego RI zmianie ulega rozmiar przekazanego mu pakietu. Realizacja programowa węzła RI IPv6 nie obsługuje nagłówków Ethernet. Do przetwarzania przekazywany jest mu wyłącznie pakiet IPv6 pozbawiony wszystkich nagłówków niższych warstw. Z tego względu po przetworzeniu każdego pakietu a przed jego wysłaniem wymagane jest zbudowanie nowych nagłówków Ethernet i IIP przez warstwę L1/2 systemu IIP.

### 3.3 Przetwarzanie ramek na poziomach fizycznym i wirtualizacji - ścieżka wyjściowa

Obsługa ramek na ścieżce wyjściowej składa z czterech etapów: przetwarzania na porcie wirtualnym, klasyfikacji ramki do odpowiedniego portu fizycznego, zarządzania ruchem oraz przetwarzania ramki na porcie fizycznym. Na Rys. 4b został przedstawiony uproszczony schemat przetwarzania ramki na poziomie Wirtualizacji. Przetwarzanie zostało zrealizowana na procesorach TOPresolve, TOPsearchII, TOPmodify oraz w module Traffic Manager. Pierwszym etapem obsługi ramki jest przetwarzanie ramki na wirtualnym porcie wyjściowym. Dla każdego portu wirtualnego określone jest jakie nagłówki mają zostać dodane do danych otrzymanych z Równoległych Internetów. Na wyjściowym porcie wirtualnym możliwe są następujące akcje:

- dodanie nagłówka Ethernet-owego,
- dodanie pola nagłówka 802.1Q,
- dodanie nagłówka PI,
- ustawienie pola Vlan w nagłówku 802.1Q.

Drugim etapem obsługi ramki jest klasyfikacja ramki do portu fizycznego, który polega na przełączaniu ramki do odpowiedniej kolejki skojarzonej z portem fizycznym. Następnie ramka zostaje przekazana do modułu Traffic Manager gdzie: 1) może zostać poddana mechanizmom zarządzania ruchem na poziomie Równoległych Internetów (mechanizmy zależne od Równoległego Internetu), 2) zostaje poddana mechanizmom zarządzania ruchem na poziomie Wirtualizacji, których celem jest podział przepustowości łącza fizycznego pomiędzy łącza wirtualne. Mechanizm szeregowania pakietów zaproponowany w projekcie IIP[1] nie może zostać zrealizowany na urządzeniu EZappliance ponieważ to urządzenie nie dostarcza funkcji szeregowania pakietów w oparciu o cykl. W konsekwencji, podział przepustowości łącza fizycznego został zrealizowany z wykorzystaniem kombinacji dwóch mechanizmów: mechanizmu szeregowania pakietów typu Deficit Round Robin (DRR), oraz mechanizmu kształtowania profilu strumienia pakietów typu Token Bucket. Mechanizm DRR zapewnia, że każdy Równoległy Internet otrzyma przynajmniej przydzieloną mu część przepływności, natomiast




**Rysunek 5.** Testowanie mechanizmu szeregowania pakietów.

mechanizm TB nie pozwala by dany RI wykorzystał więcej przepływności niż została mu przydzielona. Kombinacja mechanizmów DRR oraz TB została dokładnie przebadana w celu sprawdzenia czy działają one zgodnie z oczekiwaniami. W wyniku testów potwierdzono, że mechanizmy te działają prawidłowo zarówno w długich jak i krótkich skalach czasu. Na Rys. 5 został przedstawiony wynik przykładowego testu, wykres przedstawia szybkość przenoszonego ruchu dla poszczególnych RI w funkcji czasu (okno, w którym wyliczano szybkość ruchu wynosi 1ms). W teście założono, że ramki należące do trzech Równoległych Internetów napływają z szybkością 1Gb (długość ramki 512B). Mechanizmy ustawiono tak, aby ograniczały szybkość dostępną dla każdego RI do 300mb.

## 4 Wirtualizacja warstwy sterowania

System IIP umożliwiający koegzystencje kilku równolegle działających węzłów sieci Równoległych Internetów wymaga niezależnego działania warstw sterowania kontrolujących procesy przetwarzania pakietów w obrębie każdego z wirtualnych węzłów. Każda warstwa sterowania potrzebuje mechanizmów dostępu do wybranych struktur pamięciowych warstwy transmisji danych skojarzonych z danym wirtualnym węzłem. Istnienie kilku wirtualnych węzłów na pojedynczym fizycznym węźle sieci wiąże się z koniecznością zarządzania wirtualizacją węzła oraz dzieleniem zasobów fizycznych pomiędzy wszystkie wirtualne węzły.

| Funkcjonalność | Zarządzane obiekty procesora | Udostępnione metody | Opis |
|---|---|---|---|
| Odczyt/Zapis Struktur | TopParse, TopSearch | get, set, get_result, del_all_entries, get_limit, get_length | Grupa funkcji operująca na strukturach zapisanych w procesorze sieciowym. Możliwy jest pełen dostęp do zapisu i odczytu poszczególnych struktur. |
| Pamięć/Rejestry | Creg | set_CREG_class get_CREG_class | Grupa funkcji operująca na rejestrach procesora sieciowego umożliwiająca zapis i odczyt. |
| Konfiguracja TM | WFQ, Shaping, WRED Queue priority | set_shaping_profile, get_shaping_profile, set_entity, get_entity, set_WFQ_profile, get_WFQ_profile, set_WRED_profile, get_WRED_profile | Grupa funkcji operujących na obiekcie TM. Umożliwia konfigurowanie wybranych mechanizmów, zapis i odczyt parametrów. |
| Statystyczne | Liczniki | set_TokenBucket_profile, set_TokenBucket_counter, get_TokenBucket_profile, get_TokenBucket_counter | Grupa funkcji operująca na licznikach. Pozwala odczytać i zapisać stan wcześniej zadeklarowanych elementów. |
| Obsługa ramek | Pamięć ramki | frameSend frameRead | Grupa funkcji operująca na pamięci ramki procesora sieciowego. Umożliwia wysyłanie i odbieranie ramek bezpośrednio z procesora sieciowego. |

**Tablica 1.** Funkcje API procesora sieciowego NP-3 oferowane przez moduł EZdriver API Proxy.

### 4.1 Dostęp do warstwy transmisji danych

W celu zapewnienia warstwie sterowania dostępu do warstwy transmisji danych, został zaimplementowany moduł pośredniczący EZdriver API Proxy. Moduł ten wykorzystuje, dostarczoną przez producenta procesora sieciowego NP-3, bibliotekę EZdriver API służącą do obsługi interfejsu PCI (patrz Rys. 2) i udostępnia funkcjonalność tej biblioteki w postaci serwisu sieciowego opartego o technologię Corba. Protokół technologii Corba jest protokołem binarnym z niewielkim narzutem danych, pozwalającym zdalnie wywoływać procedury. Interfejs komunikacyjny stworzony w oparciu o tą technologię jest dostępny w sieci lokalnej węzła i może być używany przez oprogramowanie warstwy sterowania uruchomionej na zewnętrznym serwerze PC. Moduł pośredniczący EZdriver API Proxy został zaimplementowany w języku C. Moduł ten udostępnia funkcje API procesora NP-3 i pozwala kontrolować procesy obsługi ramek sieciowych zachodzące w jego wnętrzu. Dostępne metody API procesora zostały opisane w postaci specyfikacji IDL, którą potrzebuje każdy klient tego interfejsu. Zdalne wywoływanie procedury API procesora pozwala aplikacjom klienckim na operacje zapisu i odczytu bezpośrednio z/do procesora bez potrzeby posiadania własnej instancji bibliotek EZdriver API. Głównymi użytkownikami interfejsu dostępu do procesora sieciowego są: agent zarządzania zasobami fizycznymi i wirtualizacją węzła oraz moduły sterowania wirtualnymi węzłami Równoległych Internetów. Funkcje API, pogrupowane w bloki zgodnie z dokumentacją techniczną procesora NP-3 [4], są przedstawione w Tabeli 1.

Podstawowwymi funkcjami API procesora sieciowego NP-3 są operacje na strukturach danych przynależnych do procesorów TOP: TopParse i TopSearch, oraz operacje na rejestrach CREG i CAM. Powyższe struktury danych są wykorzystywane zarówno przez agenta zarządzania wirtualizacją węzła (np.: sterowanie procesem klasyfikacji przychodzących do urządzenia pakietów) jak i przez moduły sterowania wirtualnych węzłów Równoległych Internetów (np.: zarządzanie wpisami w tablicach rutingu, tablicach przełączania, etc). Trzecią grupą funkcji API są metody pozwalające konfigurować moduły TM tak, aby zapewnić prawidłowe kształtowanie wychodzącego z urządzenia ruchu poszczególnych Równoległych Internetów oraz zapewnić założony poziom jakości usług dla przetwarzanych pakietów (ustawienie właściwych paramentów mechanizmów szeregowania oraz kształtowanie ruchu). Operowanie na elementach statystycznych umożliwia kolejna grupa funkcji API. Funkcje statystyczne mogą być wykorzystywane przez agenta zarządzania do monitorowania ruchu danych na poziomie portów fizycznych urządzenia lub poziomie interfejsów wirtualnych węzłów. Dodatkowo te funkcje są wykorzystywane przez Równoległy Internet IPv6 QoS w celu kontroli prędkości bitowej na potrzeby mechanizmów QoS. Ostatnią grupą funkcji API procesora sieciowego jest przesyłanie ramek pomiędzy warstwą sterowania a procesorem sieciowym w obrębie pojedynczego węzła Systemu IIP. Mechanizm ten jest wykorzystany np. przez warstwę sterowania IPv6 QoS do rozgłaszania ramek OSPF pomiędzy węzłami sieci IPv6 QoS. Wykorzystanie modułu pośredniczącego pozwala uzyskać elastyczność oraz ujednolicić dostęp do procesora sieciowego NP-3, czyli zwirtualizowanej warstwy transmisji da-

nych, z dowolnego systemu sterowania lub zarządzania węzłem. Jednocześnie zastosowanie modułu pośredniczącego zwalnia deweloperów warstwy sterowania z konieczności posiadania ścisłej wiedzy dotyczącej działania niskopoziomowego oraz metodologii programowania procesora sieciowego oraz dostarczonych bibliotek API procesora. Dzięki zastosowaniu interfejsu opartego na technologii Corba aplikacje klienckie uniezależniają się od języka programowania, w jakim dostarczona jest biblioteka do obsługi procesora sieciowego (język C). Deweloper aplikacji klienckiej skupia się na implementacji kluczowej logiki, a do komunikacji z procesorem sieciowym wykorzystuje jednolity interfejs komunikacyjny.

### 4.2 Zarządzanie wirtualizacją węzła

Zarządzanie zasobami sieciowymi przez operatora jest bardzo ważnym aspektem prawidłowego działania sieci. Zazwyczaj nie jest łatwym zadaniem ręczne panowanie nad inwentaryzacją sieci, jej konfiguracją oraz monitorowaniem prawidłowości działania. W sieci zwirtualizowanej jest ono jeszcze bardziej skomplikowane, ponieważ oprócz zarządzania zasobami fizycznymi dochodzi również zarządzanie zasobami wirtualnymi. Nie bez znaczenia jest również fakt, że zasoby jak i ich sposób konfiguracji na poszczególnych platformach sprzętowych są mocno zróżnicowane.

```
<interfaces>
  <interface name="1">
    <v_port name="1">
      <pi type="QOS">
        <VLAN-TAG>134</VLAN-TAG>
        <PIH>010</PIH>
      </pi>
      <SourceAddressfiltering>
        bd:e3:3e:a4:23:bb
      </SourceAddressfiltering>
      <DestinationAddressfiltering>
        ad:e3:3e:a4:24:bb
      </DestinationAddressfiltering>
      <VPTIin>mac</VPTIin>
      <PIPayloadFlag>0</PIPayloadFlag>
      <VirtualPortType>0x02</VirtualPortType>
      <outVQI name="1"><PQI>1</PQI></outVQI>
      <outVQI name="2"><PQI>2</PQI></outVQI>
    </v_port>
    <Scheduler type="EZ">
      <Weight pi="QOS">50</Weight>
      <Weight pi="CAN">100</Weight>
      <CIR pi="QOS">200000</CIR>
      <CBS pi="QOS">1024</CBS>
      <CIR pi="CAN">400000</CIR>
      <CBS pi="CAN">2048</CBS>
    </Scheduler>
    <Classification>PPI</Classification>
  </interface>
</interfaces>
```

**Listing 1.** Przykładowy plik konfiguracyjny węzła systemu IIP dla platformy EZappliance.

Aby uprościć proces zarządzania w Systemie IIP, zaimplementowany został system zarządzania pozwalający na konfigurowanie zasobów w sposób jednolity

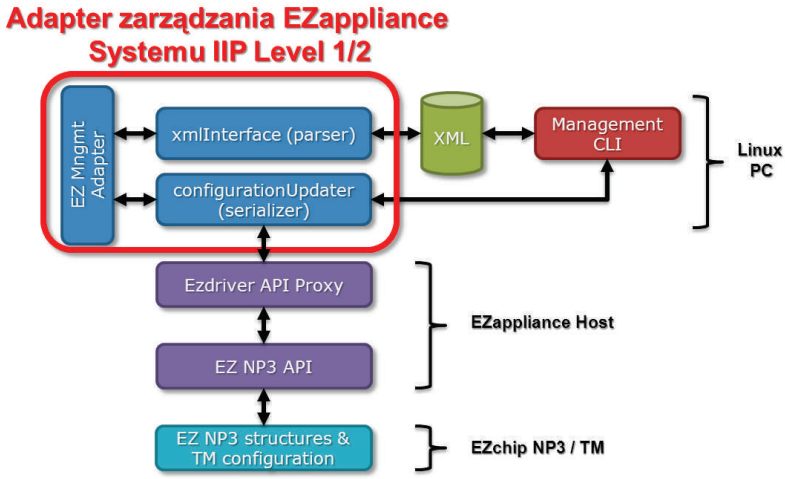
| Zarządzany element | Funkcjonalność | Opis |
|---|---|---|
| Port wirtualny (dla każdego z Równoległych Internetów) | Konfiguracja nagłówka Równoległego Internetu | Dla każdego RI można ustawić wartości nagłówka RI. |
| | Konfiguracja przetwarzania ramki na wejściowym porcie wirutalnym | Dla każdego portu wirtaulnego można skonfigurawać jaka część ramki ma zostać przełączona do węzła RI oraz czy przychodzące ramki mają być filtrowane |
| | Konfiguracja przetwarzania ramki na wyjściowym porcie wirtualnym | Dla ramki wychodzącej można dodać nagłówki: Ethernet, 8021.Q oraz PI |
| Mechanizm szerogowania pakietów (ang. scheduler) dzielący pasmo pomiędzy RI | Przydział pasma dla poszczególnych Równoległych Internetów | Dopuszczalny ruch można określić poprzez parametry:<br>– Committed Information Rate (CIR) - maksymalna szybkość bitowa,<br>– Commited Burst Size (CBS) - maksymalna wielkość paczki danych,<br>– Weight - waga dla mechanizmu szeregowania ruchu. |

**Tablica 2.** Zarządzane elementy systemu wirtualizacji na urządzaniu EZappliance.

dla różnych platform sprzętowych implementujących wirtualizację węzła Systemu IIP. Aby taki proponowany system mógł współpracować z różnymi platformami sprzętowymi potrzebny był wspólny model parametrów oraz stworzenie adapterów styków zarządzania dla każdej z platform. Adapter styku zarządzania dla platformy EZappliance to sterownik potrafiący konfigurować sprzęt jak i logika, która w odpowiedni sposób pozwoli na wykonanie takich czynności konfiguracyjnych, aby spełnić wymagania Systemu Zarządzania IIP. Ponieważ pomiędzy poszczególnymi platformami wykorzystanymi w projekcie IIP występuje znaczne zróżnicowanie, to też model parametrów systemu zarządzania zawiera pewne specyficzne tylko dla EZappliance parametry. Wprowadzane nowe parametry lub modyfikacje obecnych wartości zapisywane są w pliku konfiguracyjnym XML. Na polecenie administratora węzła, adapter wczytuje konfigurację i wykonuje proces ustawiania na urządzeniu. Daje to dodatkową możliwość samodzielnego tworzenia lub zmiany bezpośrednio w pliku XML. Przykład wyglądu pliku konfiguracji węzła systemu IIP dla platformy EZappliance przedstawiono w Listing 1.

Wirtualizacja węzła Systemu IIP może być zarządzana przez administratora poprzez interfejs linii komend (CLI) oraz poprzez graficzny interfejs Systemu Zarządzania IIP dostępny poprzez przeglądarkę internetową. Aktualna implementacja adaptera zarządzania dla EZappliance pozwala na konfigurowanie parametrów przedstawionych w Tabeli 2.

Moduł adaptera zarządzania dla EZappliance zaimplementowany został, jako aplikacja w języku Python. Adapter zarządzania ustawia żądane parametry wykorzystując do tego interfejs oferowany przez moduł pośredniczący EZdriver API Proxy. Adapter umożliwia również odczyt z urządzenia wartości parametrów. Rys. 6 przedstawia schemat budowy systemu zarządzania wirtualizacją węzła.




**Rysunek 6.** Budowa adaptera zarządzania dla platformy EZappliance.

## 5 Podsumowanie

EZappliance jest programowalną platformą opartą o procesor sieciowy NP-3 i zaprojektowaną do łatwego wdrażania nowych funkcjonalności sieciowych. Artykuł przedstawia jak platforma EZappliance została użyta do stworzenia implementacji węzła Systemu IIP oraz opisuje szczegóły zaimplementowanych mechanizmów wirtualizacji zasobów sieciowych zarówno w warstwie transmisji danych jak i warstwie sterowania węzłem. Przeprowadzone testy i demonstracje działania węzła Systemu IIP opartego o tą platformę pokazują, że jest to rozwiązanie wydajnie przetwarzające pakiety niezależnie od aktualnego obciążenia ruchem portów wejściowych urządzenia, co jest możliwe dzięki budowie procesora NP-3 zawierającego wiele równoległych potoków sprzętowo akcelerowanego przetwarzania pakietów sieciowych. Niestety, taka budowa procesora sieciowego, dostosowana przede wszystkim do aktualnie stosowanych technologii sieciowych, stanowiła jednocześnie utrudnienie dla innowacji, czyli wyodrębnienia niezależnych węzłów wirtualnych i implementacji nowych algorytmów szeregowania pakietów. Dodatkowo w czasie implementacji węzła Systemu IIP okazało się, że platforma

EZappliance posiada zbyt mało wydajny system z procesorem ogólnego przeznaczania (Host CPU), aby uruchamiać programy napisane w językach innych niż C/C++.

## Literatura

1. W. Burakowski, et al. Architektura Systemu IIP, Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź, 14-16 września 2011
2. W. Burakowski (Ed), Specyfikacja systemu IIP - poziom 1 i 2 - wersja 1, Raport projektu "Inżynieria Internetu Przyszłości", POIG.01.01.02-00-045/09-00, 2011
3. W. Burakowski , H. Tarasiuk, A. Bęben, System IIP for supporting "Parallel Internets (Networks)", Future Internet Assembly meeting, Ghent, 2010, http://fi-ghent.fi-week.eu/files/2010/12/1535-4-System-IIP-FIA-Ghent-ver1.pdf
4. EZchip Technologies, EZappliance product
5. EZchip Technologies, http://ezchip.com

# Realizacja węzła Systemu IIP z wykorzystaniem Xen

Błażej Adamczyk, Andrzej Chydziński

Instytut Informatyki
Wydział Automatyki, Elektroniki i Informatyki
Politechnika Śląska

**Streszczenie** Jedna z trzech platform, na której zrealizowane zostały węzły Systemu IIP, jest oparta na programowej wirtualizacji, wykorzystującej znany monitor maszyn wirtualnych Xen. Platformę taką można uruchomić na dowolnym komputerze wspierającym wirtualizację (technologie Intel VT-x oraz AMD-V). Przedstawiamy szczegółowo implementację węzła Systemu IIP na platformie Xen oraz wstępne wyniki testów funkcjonalności i wydajności tej platformy.

## 1 Wprowadzenie

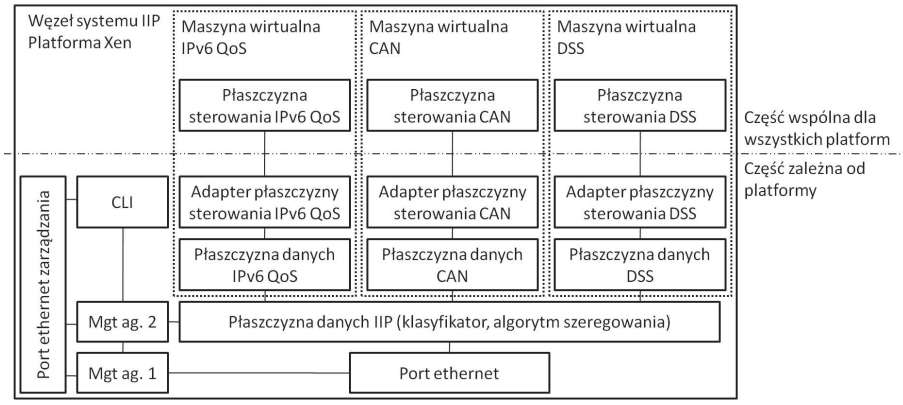
Na całym świecie naukowcy intensywnie pracują nad koncepcjami Internetu Przyszłości. W Polsce prace te prowadzone są w ramach projektu Inżynieria Internetu Przyszłości (IIP) [1,2], którego jednym z najważniejszych rezultatów będzie prototypowa architektura, nazwana Systemem IIP [3]. Wprowadza ona warstwę wirtualizacji udostępniającej wirtualne (zamiast fizycznych) węzły i łącza. Takie rozwiązanie pozwala na zbudowanie, na wspólnej infrastrukturze fizycznej, wielu Równoległych Internetów (Parallel Internets - PI), o dowolnej topologii i różnych mechanizmach transmisji danych. Trzy Równoległe Internety zostały zaproponowane w Systemie IIP: IPv6 QoS – sieć z gwarancją jakości usług i protokołem IPv6, CAN (Content Aware Network) – sieć świadoma przekazywanej treści oraz DSS (Data Stream Switching) – sieć przełączająca strumienie danych.

Do implementacji Systemu IIP wykorzystano trzy platformy. Te platformy to: programowa - wykorzystująca monitor maszyn wirtualnych Xen, oraz dwie sprzętowe, wykorzystujące odpowiednio układ programowalny NetFPGA i programowalny procesor sieciowy EZchip NP-3 (więcej informacji na temat tych platform - zobacz [11]). W celu ujednolicenia koncepcji węzła Systemu IIP, realizowanej na tych różnych platformach, stworzony został model idealnego urządzenia umożliwiającego wirtualizację [4].

Artykuł niniejszy przedstawia szczegóły implementacyjne platformy programowej, opartej na Xen. Jej główną zaletą jest fakt, że można ją uruchomić na dowolnym komputerze PC umożliwiającym wirtualizację. Co więcej, pozwala ona na wykorzystanie dowolnego systemu operacyjnego z wszystkimi jego funkcjami, co znacznie ułatwia implementację mechanizmów wyższych warstw (np. warstwy sterowania) poprzez użycie już istniejących funkcjonalności.

Dalszy układ artykułu przedstawia się następująco. W rozdziale 2 przedstawiony jest ogólny model implementacyjny węzła Systemu IIP. Rozdział 3 opisuje najbardziej istotne tutaj aspekty środowiska wirtualizacji Xen. W rozdziale 4 pokazane są problemy, które pojawiły się w pracach nad implementacją węzła Systemu IIP na platformie Xen. W rozdziale 5 pokazana jest szczegółowa implementacja klasyfikatora i mechanizmu szeregowania. Z kolei w rozdziale 6 przedstawione są wyniki wstępnych eksperymentów wykonanych w celu weryfikacji poprawności działania oraz oszacowania wydajności platformy Xen. W rozdziale 7 znajduje się podsumowanie artykułu.

## 2 Ogólny model węzła Systemu IIP




**Rysunek 1.** Schemat komunikacyjny węzła Systemu IIP dla platformy Xen.

Jak już wcześniej wspomniano, fizyczne węzły Systemu IIP zostały zbudowane z wykorzystaniem trzech różnych platform. Platformy te różnią się zastosowanym sprzętem, więc implementacja na nich musi również się różnić. Aby jednak mechanizmy wyższych warstw mogły być niezależne od platformy, model Systemu IIP [5] zakłada podział węzła na 2 moduły: pierwszy, zależny od zastosowanej platformy oraz drugi, niezależny. Aby zapewnić wysoką wydajność, funkcje przekazu danych (Data Plane - DP), wchodzą w skład pierwszego modułu. Funkcje płaszczyzny sterowania (Conrol Plane - CP), ze względu na niski wpływ na wydajność systemu, są zaimplementowane we wszystkich platformach tak samo, niezależnie od sprzętu. Te ostatnie, w przypadku platform sprzętowych (NetFPGA oraz EZchip), zostały przeniesione do oddzielnej instalacji systemu Xen, gdzie funkcje specyficzne dla różnych Równoległych Internetów wykonywane są w odrębnych maszynach wirtualnych. W przypadku czysto programowej platformy Xen, nie jest to konieczne, ponieważ wspomniane maszyny wirtualne

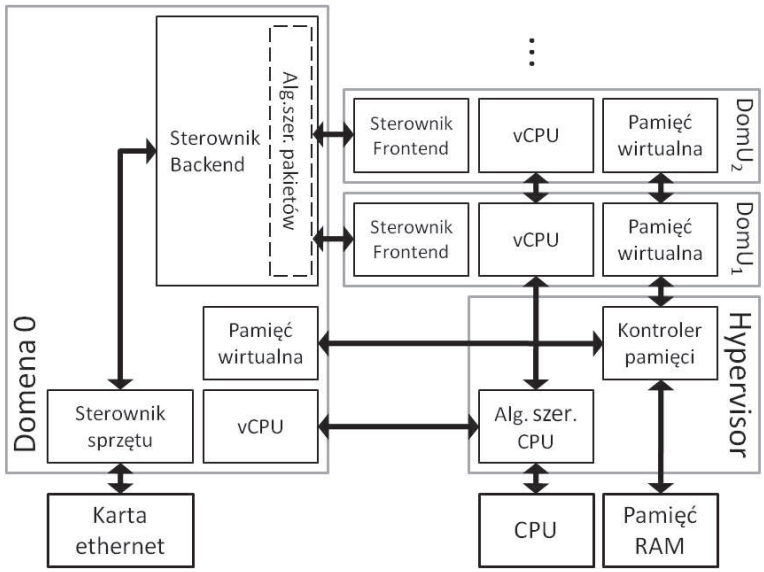
mogą być uruchomione na tym samym sprzęcie, obsługującym DP. Rys. 1 przedstawia architekturę węzła Systemu IIP z uwzględnieniem płaszczyzn CP i DP oraz podziałem na wspomniane moduły.

Jak widać, w platformie Xen, zarówno płaszczyzna DP jak i CP jednego Równoległego Internetu znajdują się wewnątrz maszyny wirtualnej. W takiej strukturze każda maszyna stanowi kompletny, osobny węzeł wirtualny, z wirtualnym łączem. Pozwala to na korzystanie z istniejącego stosu protokołów wirtualizowanego systemu operacyjnego, co ułatwia proces implementacji.

Oprócz warstwy DP na poziomie Równoległych Internetów, wyróżnić można również wspólną płaszczyznę przekazu danych, w skład której wchodzą: mechanizm klasyfikacji oraz algorytm szeregowania określony w [4]. Te funkcjonalności są również mocno związane z zastosowaną platformą i dlatego zostały zaimplementowane osobno.

## 3 Środowisko wirtualizacji Xen

Wirtualizacja systemów komputerowych jest obecnie szeroko stosowana. Do tej pory powstało już wiele różnych monitorów maszyn wirtualnych. Xen jest jednym z bardziej popularnych, a w dodatku otwartym (Open Source) środowiskiem wirtualizacji. Dzięki swojej specyficznej architekturze (patrz Rys. 2 oraz [6]) jest jednocześnie wydajny i niezawodny. Dlatego to właśnie Xen, został wybrany (po dogłębnej analizie) na potrzeby wirtualizacji węzłów i łączy w projekcie IIP.




**Rysunek 2.** Schemat komunikacyjny węzła Systemu IIP dla platformy Xen.

Zrozumienie szczegółów implementacji węzła Systemu IIP na platformie Xen wymaga dobrego zrozumienia zasady działania różnych mechanizmów tego środowiska. Najważniejszym elementem Xen jest hypervisor – główny system operacyjny uruchomiony bezpośrednio na sprzęcie, bez restrykcji w kręgu zabezpieczeń 0 (ang. Ring0). Zadaniem hypervisora jest wirtualizacja kluczowych zasobów tj. procesora oraz pamięci RAM. Bezpośrednio po starcie hypervisor uruchamia pierwszą maszynę wirtualną, zwaną Domeną0, w bardziej restrykcyjnym kręgu zabezpieczeń (Ring1w przypadku architektury x86 lub Ring3 w przypadku x86_64). Domena 0 jest jedną z uprzywilejowanych maszyn wirtualnych, co oznacza, że ma bezpośredni dostęp do niektórych urządzeń wejścia/wyjścia, takich jak karty sieciowe lub dyski. Dopiero uprzywilejowana domena może udostępniać urządzenia następnym maszynom - jest ona wtedy nazywana domeną sterowników (ang. driver domain). Takie współdzielenie urządzeń jest osiągane za pomocą specjalnych sterowników parawirtualnych (ang. paravirtual drivers). Są to dwa specjalne rodzaje sterowników: sterowniki Backend – uruchomione w domenie 0, odpowiedzialne za komunikację z prawdziwym sterownikiem oraz udostępnianie zasobu innym domenom; oraz sterowniki Frontend – uruchomione w pozostałych maszynach, odpowiedzialne za komunikację ze sterownikiem Backend oraz udostępnianie wirtualnego zasobu w systemie maszyny wirtualnej. Użycie istniejących już sterowników (nie parawirtualnych) jest również możliwe w Xen, ale wymaga zastosowania emulacji, co znacznie zmniejsza wydajność takiego rozwiązania.

Taka architektura ma niewątpliwie wiele istotnych zalet. Przede wszystkim, zakres obowiązków głównego systemu operacyjnego jest zminimalizowany, co w znacznym stopniu poprawia stabilność i bezawaryjność. Urządzenia wejścia-/wyjścia są obsługiwane przez maszyny wirtualne, co pozwala na odizolowanie różnych sterowników (ich ewentualna awaria nie wpływa na pracę innych maszyn oraz hypervisora). Dodatkowo, sterowniki te mogą być instalowane w różnych wirtualnych systemach operacyjnych, co sprawia że zdecydowana większość urządzeń jest wspierana przez architekturę Xen.

Kluczową rolę w procesie wirtualizacji pełnią mechanizmy szeregowania. To dzięki nim można określać sposób w jaki współdzielone są zasoby. Algorytm szeregowania pakietów Systemu IIP [4] został zaprojektowany z myślą o izolacji i jakości usług Równoległych Internetów. W architekturze Xen, algorytm ten jest zaimplementowany wewnątrz sieciowego sterownika Backend, w wątku jądra o nazwie Netback. Wbudowany w Xen oryginalny algorytm to bardzo prosty tzw. credit scheduler, który przydziela kolejnym wirtualnym interfejsom sieciowym (vif) kredyty, a te z kolei zużywają je wysyłając pakiety. W pierwszej kolejności przetwarzane są żądania tych domen, które nie wykorzystały wszystkich kredytów. W ten sposób, w dużym przybliżeniu osiągany jest równy podział pasma pomiędzy wirtualnymi interfejsami. Na potrzeby Systemu IIP algorytm ten musiał być zmieniony w celu zapewnienia znacznie większej kontroli i izolacji wirtualnych łączy.

# 4 Problemy związane z wirtualizacją

Już wstępne badania wykazały kilka problemów związanych z wirtualizacją, których rozwiązanie było niezbędne aby implementacja Systemu IIP w platformie Xen była możliwa. W kolejnych podrozdziałach przedstawione zostaną trzy takie problemy oraz sposoby ich rozwiązania.

## 4.1 Wydajność wirtualnych interfejsów sieciowych

Łączna przepustowość osiągana na łączach wirtualnych powinna być maksymalnie zbliżona do przepustowości fizycznego interfejsu sieciowego. Wstępne testy wykazały jednak, że nie łatwo jest to osiągnąć już przy łączach o przepustowości 1Gb/s, a zwłaszcza w środowiskach wirtualnych [7]. Wpływ na ten efekt może mieć kilka czynników. Przede wszystkim, w przypadku gdy więcej jest wirtualnych procesorów niż fizycznych, lub ich asocjacja jest niepoprawna, algorytm szeregowania CPU może powodować opóźnienia w obsłudze karty sieciowej, a co za tym idzie - utratę pakietów i stratę przepustowości. Ten problem może zostać zminimalizowany poprzez asocjację procesów związanych z obsługą wirtualnego interfejsu z jednym wirtualnym procesorem, a następnie powiązanie tego wirtualnego procesora z konkretnym procesorem fizycznym.

Kolejną przyczyną niedostatecznej wydajności wirtualnych interfejsów może być obsługa przerwań na poziomie uprzywilejowanych maszyn wirtualnych. Każde przerwanie może być obsługiwane przez wybrany procesor. Rozłożenie najbardziej kluczowych przerwań (przede wszystkim przerwań interfejsu sieciowego) powinno być optymalizowane dynamicznie wraz ze zmieniającym się obciążeniem systemu. W systemach Linux odpowiedzialny jest za to proces irqbalance. Niestety, w przypadku wirtualizacji, balansowanie przerwań z poziomu maszyny wirtualnej nie jest wystarczające, ponieważ algorytm szeregowania CPU może dynamicznie zmieniać przypisania procesorów wirtualnych według ich obciążenia, co zakłóca balansowanie przerwań. Rozwiązaniem tego problemu jest, po raz kolejny, odpowiednia konfiguracja asocjacji wirtualnych procesorów oraz uruchomienie irqbalance jednorazowo, w momencie największego obciążenia interfejsów sieciowych. Jeszcze korzystniejsze (od dynamicznego irqbalance) okazuje się ręczne skonfigurowanie asocjacji przerwań tak, aby przerwania kart sieciowych były obsługiwane przez najmniej obciążone procesory wirtualne.

## 4.2 Izolacja wirtualnych interfejsów sieciowych

Kolejnym problemem związanym bezpośrednio z mechanizmem wirtualizacji węzłów i łączy jest brak wystarczającej izolacji wydajnościowej pomiędzy interfejsami wirtualnymi [8,9]. W niezmodyfikowanej instalacji Xen aktywność jednego wirtualnego interfejsu może mieć znaczny wpływ na wydajność i działanie innych. Przy implementacji Systemu IIP efekt ten jest oczywiście niepożądany i powinien być zminimalizowany. Algorytm szeregowania zaproponowany dla Systemu IIP był projektowany z myślą o izolacji wirtualnych łączy, dlatego poprawia on istotnie izolację bez innych modyfikacji. Izolację wydajnościową można

jeszcze dodatkowo poprawić stosując odpowiednie reguły filtracji bezpośrednio na ruchu wchodzącym do węzła fizycznego.

### 4.3 Generowanie opóźnień czasowych wysokiej rozdzielczości

Wspomniany wcześniej algorytm szeregowania Systemu IIP wymaga mechanizmu generowania opóźnień czasowych o wysokiej rozdzielczości. Długość fazy pojedynczego Równoległego Internetu, w zależności od zakładanych opóźnień i przepustowości, może zaczynać się już od dziesiątek mikrosekund. Niestety, w oryginalnym Xen, mimo włączenia obsługi hrtimers w jądrze maszyny wirtualnej, rozdzielczość generatora opóźnień wynosi około 100us, czyli o około rząd wielkości za mało. Aby osiągnąć wymaganą rozdzielczość, niezbędna jest modyfikacja zarówno jądra domeny0, jak również samego hypervisora. Zmiana mechanizmów optymalizacji liczby generowanych opóźnień pozwala na znaczne poprawienie rozdzielczości tych generatorów kosztem dodatkowego przetwarzania. Z drugiej strony, jądro domeny0 może być uruchomione w trybie tickless, przez co można znacznie zmniejszyć liczbę przerwań czasowych.

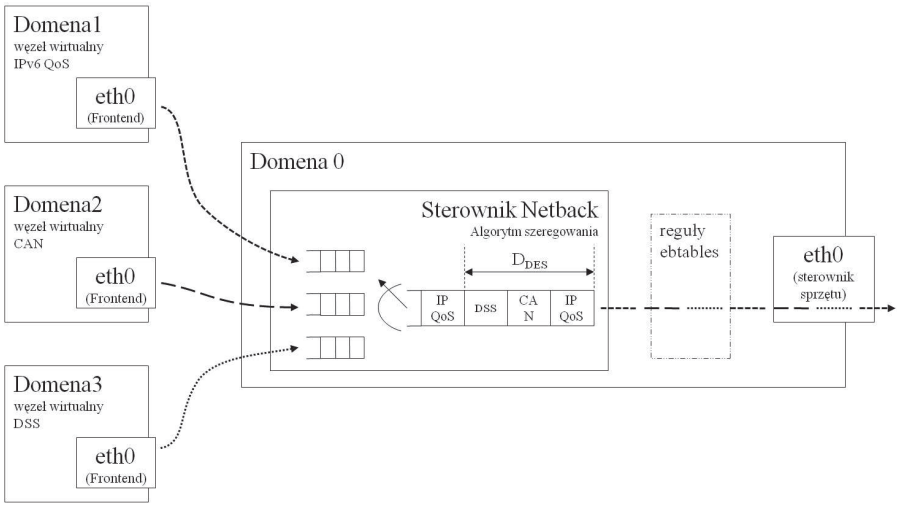
## 5 Implementacja

Czysto programowa platforma Systemu IIP jest oparta na Xen Hypervisor 4.0.0 i systemie Linux Gentoo 2.6.38-xen (zrówno w domenie0, jak i w innych maszynach wirtualnych). Zarówno klasyfikacja ruchu wejściowego, jak również szeregowanie ruchu wyjściowego zaimplementowane jest w domenie0. Jak już wspomniano, algorytm szeregowania Systemu IIP jest osadzony w miejscu oryginalnego algorytmu dostarczonego przez środowisko Xen. Klasyfikacja jest natomiast przeprowadzana poprzez dobrze znany mechanizm reguł o nazwie ebtables. W kolejnych podrozdziałach obydwa te mechanizmy zostaną szczegółowo opisane.

### 5.1 Algorytm szeregowania

Zaproponowany algorytm szeregowania dla Systemu IIP jest to algorytm z podziałem na fazy czasowe dla kolejnych Równoległych Internetów. Przykładowy schemat komunikacyjny dla ruchu wyjściowego przedstawiony jest na Rys. 3.

Niemodyfikowana instalacja Xen wprowadza dodatkową kolejkę pomiędzy sterownikami Frontend i Backend. Algorytm szeregowania dla Systemu IIP zakłada tylko kolejki na poziomie wirtualnej maszyny, czyli algorytmu szeregowania poziomu drugiego. Szeregowanie poziomu pierwszego pobiera kolejne pakiety bezpośrednio z kolejek poziomu drugiego. Biorąc pod uwagę powyższe, algorytm musiał zostać dostosowany do platformy Xen (patrz Rys. 4). Sterownik Frontend został zmodyfikowany tak, aby wysyłał pakiety tylko w obrębie swojej fazy. Dodatkowo, liczy on ilość wysłanych danych i nie wysyła ich więcej, niż fizyczny adapter potrafi wysłać w przeciągu tej fazy. Taka implementacja pozwala na wykonywanie obu tych procesów równolegle, zwiększając przy tym wydajność systemu.



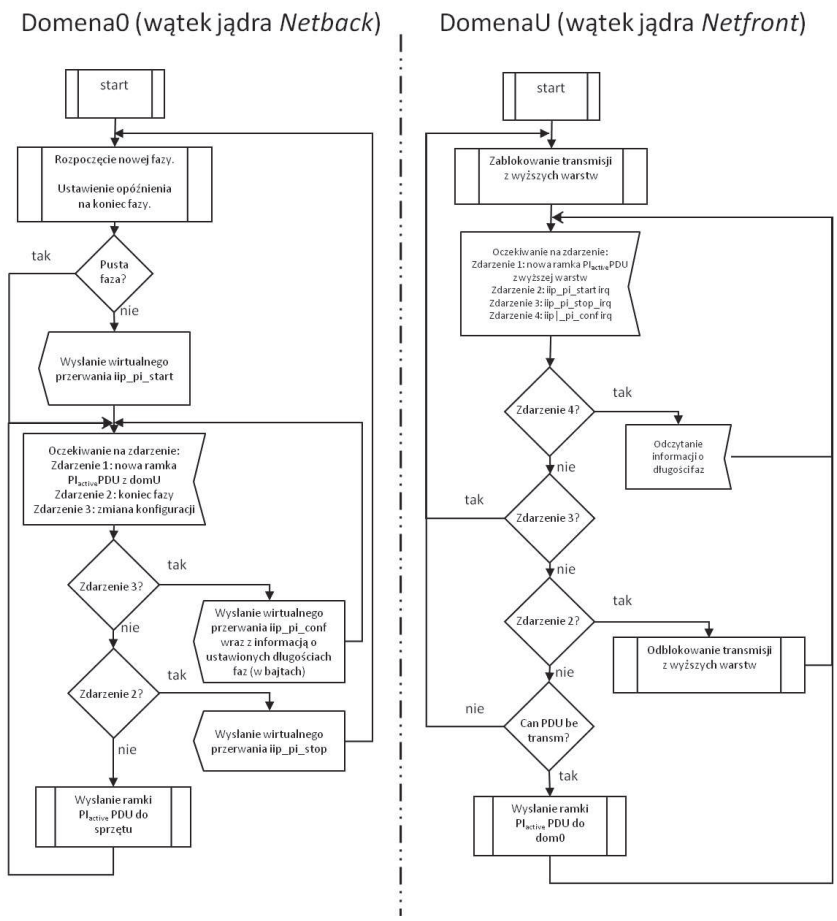

**Rysunek 3.** Schemat komunikacyjny węzła Systemu IIP dla platformy Xen.

## 5.2 Klasyfikator

Na najniższej warstwie Systemu IIP komunikacja odbywa się za pomocą protokołu Ethernet. Typ Równoległego Internetu określany jest w każdej ramce za pomocą pierwszego bajtu danych w tzw. nagłówku PI (PI header). Ruch przychodzący musi zostać sklasyfikowany i dalej przesłany do mechanizmów odpowiedniego PI.

W platformie Xen, dla każdego interfejsu fizycznego tworzony jest most sieciowy w domenie sterowników, który pozwala na współdzielenie fizycznego łącza przez kilka wirtualnych interfejsów (patrz Rys. 5). W systemach Linux mosty pozwalają na definiowanie reguł ebtables (ang. Ethernet Bridge Tables, [10]), którym poddawane są wszystkie pakiety przechodzące przez dany most. Na potrzeby klasyfikacji stworzono specjalne rozszerzenie mechanizmu ebtables (tzw. matching extension), które pozwala na definiowanie reguł dla ramek o konkretnym nagłówku PI. Te z kolei można wykorzystać aby zmienić adres docelowy MAC na adres odpowiedniej maszyny wirtualnej. W ten sposób most sieciowy automatycznie przekaże ramkę do odpowiedniego węzła wirtualnego (dlatego też wirtualne interfejsy w obrębie jednej maszyny fizycznej powinny mieć unikatowe adresy MAC).

Ramki wychodzące z węzła wirtualnego mają adres źródłowy MAC wyjściowego wirtualnego interfejsu. Jako że adresy interfejsów wirtualnych są unikatowe tylko w obrębie jednego węzła fizycznego, kolejna reguła mechanizmu ebtables została wykorzystana aby zmienić źródłowy adres MAC ramek wychodzących na adresy odpowiednich interfejsów fizycznych.

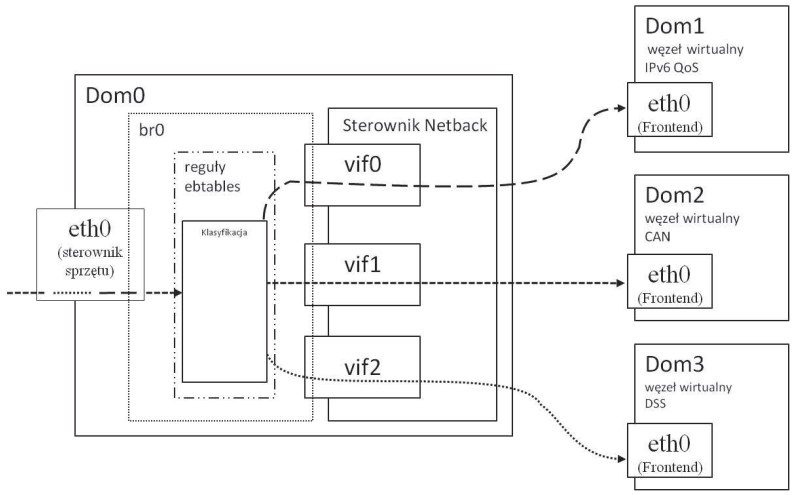



**Rysunek 4.** Algorytm szeregowania dla platformy Xen (algorytm domeny0 po lewej, algorytm domenyU po prawej).

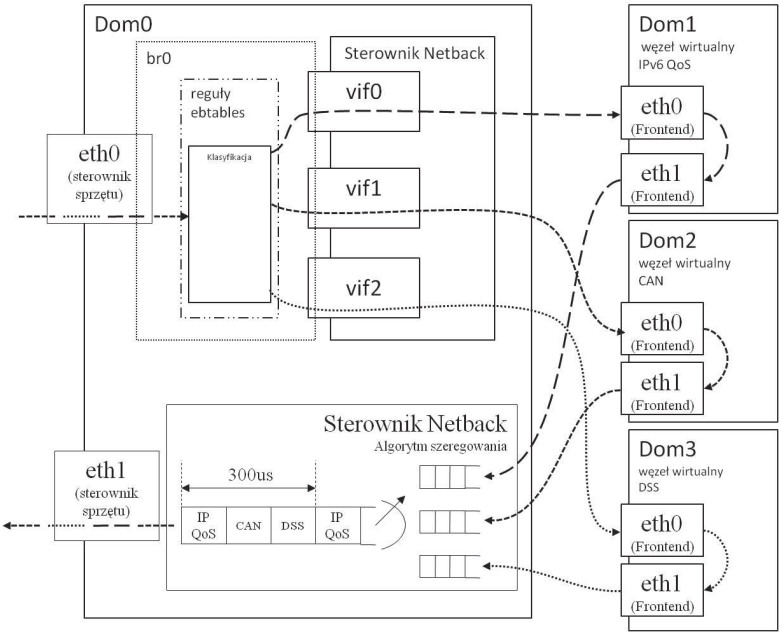
## 6 Wstępne testy

W celu weryfikacji powyżej opisanej implementacji i sprawdzenia jej wydajności przeprowadziliśmy dwa eksperymenty. Oba testy oparte były o ten sam scenariusz (patrz Rys. 6). Ruch wejściowy do węzła (port eth0) składał się z ramek o długości 1024B i generowany był za pomocą generatora Spirent SPT-2000. Ten sam sprzęt służył też do analizy danych wyjściowych węzła (port eth1).

W każdym węźle wirtualnym uruchomiona została prosta funkcjonalność CP – routing ruchu pomiędzy portami eth0 i eth1. Długość cyklu algorytmu szeregowania na porcie wyjściowym eth1 węzła fizycznego została ustawiona na 300us.




**Rysunek 5.** Mechanizm klasyfikacji pakietów Systemu IIP na platformie Xen.



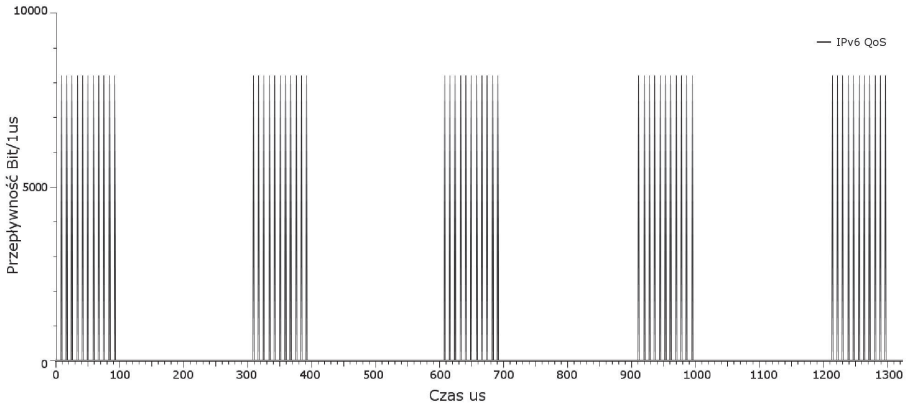

**Rysunek 6.** Konfiguracja testowa

Cykl składał się z trzech równych faz dla każdego Równoległego Internetu (po 100us).

Pierwszy eksperyment sprawdzał zachowanie węzła w przypadku aktywności tylko jednego PI, gdy przepustowość wejściowa aktywnego PI przekracza zadaną przepustowość wyjściową. Celem takiego testu jest sprawdzenie, czy na wyjściu Równoległy Internet nie przekroczy zadanej przepustowości i czy jego fazy pracy będą odpowiedniej długości, z przerwami dla pozostałych Internetów.

W drugim eksperymencie z kolei, ruch generowany był dla wszystkich trzech Internetów Równoległych. Przepustowość wejściowa była równa z oczekiwaną przepustowością wyjściową, natomiast ułożenie pojedynczych pakietów było losowe. W tej konfiguracji sprawdzana jest wydajność węzła przy pełnym obciążeniu dwóch interfejsów fizycznych oraz poprawność działania algorytmu szeregowania (ułożenie pakietów na wyjściu powinno wyraźnie wskazywać fazy różnych PI).

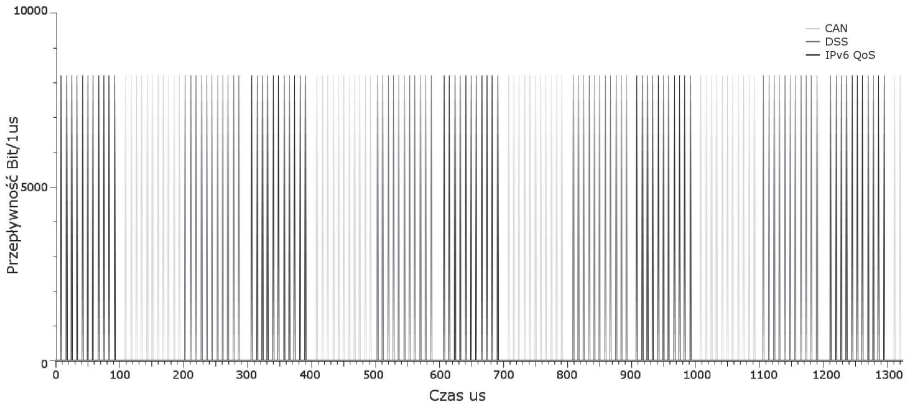
Wyniki pierwszego testu przedstawiono na Rys. 7. Ruch wyjściowy został zarejestrowany w formacie pcap, dlatego też że na wykresie nie można zaobserwować ciągłej przepływności, a raczej pojedyncze impulsy w momentach, kiedy została odebrana kompletna pojedyncza ramka. Nie przeszkadza to jednak w obserwacji sposobu działania samego algorytmu szeregowania.




**Rysunek 7.** Wykres ruchu wyjściowego węzła IIP dla aktywności jednego Internetu Równoległego.

Na wykresie dokładnie widać, że długości faz są dobrze zachowane. Fazy nieaktywnych Internetów również mają odpowiednią długość. Przepustowość osiągnięta podczas pierwszego eksperymentu była na poziomie 300Mb/s, a więc nieco niższa niż 1/3 przepustowości interfejsu fizycznego (333Mb/s). Straty te spowodowane są niedopasowaną długością fazy oraz opóźnieniami w mechanizmach generacji opóźnień czasowych. Na Rys. 8 przedstawiono wykres ruchu wyjściowego dla drugiego eksperymentu.

W drugim przypadku, również widać poprawność odmierzania faz nawet przy pełnym obciążeniu. Tutaj wyraźniej można zaobserwować straty pojedynczych




**Rysunek 8.** Wykres ruchu wyjściowego węzła IIP dla pełnej aktywności trzech Internetów Równoległych.

pakietów pod koniec faz, spowodowane niedopasowaniem długości fazy do wielkości pakietów. Przepustowość wszystkich trzech Internetów była podobna do uzyskanej w poprzednim eksperymencie, tj. 300Mb/s.

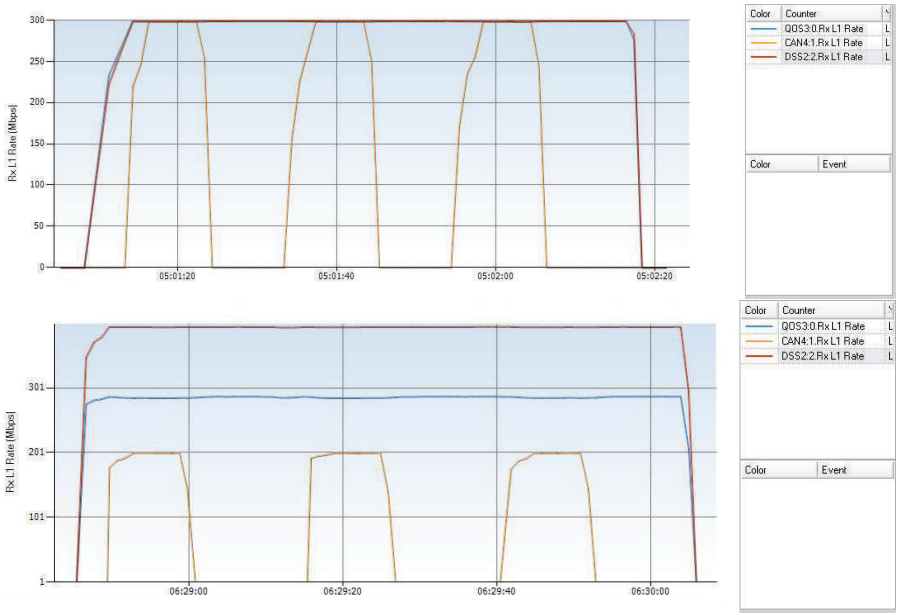
Dodatkowo przeprowadzone zostały testy na dwóch portach wyjściowych, które nie wykazały spadku wydajności. Tylko w przypadku, kiedy wykorzystane zostały dwa porty należące do tego samego wieloportowego adaptera sieciowego, można było zauważyć dodatkowy spadek przepływności. Wydaje się, że jest to spowodowane niską wydajnością tych adapterów w przypadku pełnego użycia kilku portów jednocześnie.

Przeprowadzone zostały również testy izolacji wydajnościowej równoległych internetów. Dwa z wielu przeprowadzonych pomiarów (wyniki przedstawione na rysunku 9) miały na celu sprawdzenie czy rozpoczęcie i zakończenie transmisji w jednym internecie równoległym ma wpływ na wydajność i działanie pozostałych internetów.

Pierwszy wykres na rysunku 9 przedstawia przebieg przepustowości w czasie gdzie w ramach każdego internetu równoległego generowany jest ruch przekraczający możliwości obsługi (tj. $400Mb/s$) składający się z ramek o stałej długości ($1024B$). Długość fazy każdego internetu równoległego była ustawiona na $74,3\mu s$ co według modelu teoretycznego algorytmu szeregowania powinno pozwolić każdemu internetowi równoległemu na osiągnięcie przepustowości $299,76Mb/s$. Pomiar wykazał średnią przepustowość dla każdego RI na poziomie $299,4Mb/s$.

Na drugim wykresie przedstawiono pomiar dla zbliżonego eksperymentu. Tutaj jednak dla każdego internetu generowany był ruch o innej długości ramki i innym natężeniu oraz różne rozmiary faz:

– PI CAN - ruch typu “ON/OFF” - rozmiar ramki $1470B$, natężenie $300Mb/s$, długość fazy $70\mu s$,




**Rysunek 9.** Wykresy przepustowości osiąganej przez trzy równoległe internety przedstawiający wzajemną izolację wydajnościową.

- PI IPv6 QoS - ruch CBR - rozmiar ramki $800B$, natężenie $400Mb/s$, długość fazy $90\mu s$,
- PI DSS - ruch CBR - rozmiar ramki $1024B$, natężenie $500Mb/s$, długość fazy $125\mu s$,
- Faza ciszy - $10\mu s$

Według teoretycznego modelu algorytmu szeregowania przepustowości uzyskane dla internetów: PI CAN ("ON/OFF"), PI IPv6 QoS oraz PI DSS powinny być równe kolejno: $202,03Mb/s$, $289,08Mb/s$ i $396,37$. Średnie przepustowości zmierzone były zbliżone do oczekiwanego rezultatu i kolejno wynosiły: $199,8Mb/s$, $287,3Mb/s$ i $395,4Mb/s$.

Jak widać w obu przypadkach izolacja wydajnościowa jest poprawnie zachowana, a algorytm szeregowania działa zgodnie z jego teoretycznym modelem.

## 7 Podsumowanie

W artykule przedstawione zostały szczegóły implementacyjne czysto programowej platformy wirtualizacji zasobów sieciowych dla Systemu IIP. Została ona stworzona z wykorzystaniem monitora maszyn wirtualnych Xen. Część funkcjonalności tego monitora wymagała modyfikacji, tak aby platforma Systemu IIP mogła działać poprawnie. Poza aspektami implementacyjnymi przedstawione

zostały również problemy powstałe na skutek wykorzystania technologii wirtualizacji i w skrócie opisane ich rozwiązanie. W końcu zaprezentowane zostały dwa scenariusze testów zaimplementowanej platformy wraz z wynikami. Wykorzystanie monitora maszyn wirtualnych jest pionierskim podejściem do wirtualizacji wśród projektów Internetu Przyszłości, ale jak widać jest możliwe i wydaje się być obiecujące, przynajmniej dla potrzeb prototypowania, ze względu na niższe koszty oraz łatwiejszą konfigurację.

Praca ta jest rozszerzoną wersją artykułu [12] przedstawionego na KSTiT'12, Warszawa, wrzesień 2012.

## Literatura

1. W.Burakowski, et al. Architektura Systemu IIP, Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź, 14-16 września 2011
2. W. Burakowski (Ed), Specyfikacja systemu IIP – poziom 1 i 2 – wersja 1, Raport projektu „Inżynieria Internetu Przyszłości", POIG.01.01.02-00-045/09-00, 2011
3. W. Burakowski„ H. Tarasiuk, A. Bęben, System IIP for supporting „Parallel Internets (Networks)", Future Internet Assembly meeting, Ghent, 2010, http://fi-ghent.fi-week.eu/files/2010/12/1535-4-System-IIP-FIA-Ghent-ver1.pdf.
4. W. Burakowski et al., "Idealne urządzenie umożliwiające wirtualizację infrastruktury sieciowej w Systemie IIP", KSTiT 2011, Łódź 2011
5. P. Zwierko i in., Platformy wirtualizacji dla Systemu IIP. Przegląd Telekominikacyjny nr 8-9/2011
6. D. Chisnall, The Definitive Guide to the Xen Hypervisor. Prentice Hall Press, Upper Saddle River, 2007.
7. T. Fortuna, B. Adamczyk, Improving packet reception and forwarding within virtualized Xen environments, Springer - Computer Networks: Communications in Computer and Information Science, Volume 291, 153-160, 2012
8. B. Adamczyk, A. Chydzinski, "On the Performance Isolation Across Virtual Network Adapters in Xen", Proceedings of The Second International Conference on Cloud Computing, GRIDs, and Virtualization, Rome, 2011, pp. 222-227
9. B. Adamczyk, A. Chydzinski, Performance Isolation Issues in Network Virtualization in Xen, International Journal on Advances in Networks and Services, vol. 5 no 1 & 2, pp. 139-148, 2012
10. B. De Schuymer, et al.; Ebtables;http://ebtables.sourceforge.net/.
11. A. Chydzinski i in., Virtualization Devices for Prototyping of Future Internet, Proc. of SNPD, Kyoto, August 2012. IEEE Computer Society, pp. 672-678, 2012
12. B. Adamczyk, A.Chydzinski, „Implementacja programowej platformy wirtualizacji dla Systemu IIP z wykorzystaniem monitora maszyn wirtualnych", materiały KSTiT'12, str. 1399–1408, Warszawa, wrzesień 2012.

# Współpraca systemu zarządzania z modułem wymiarowania zasobów w Systemie IIP

Janusz Granat[1], Jakub Bielecki[1], Wojciech Szymak[1],
Krzysztof Wajda[2], Mirosław Kantor[2], Janusz Gozdecki[2],
Jacek Rak[3]

[1] Instytut Łączności - Państwowy Instytut Badawczy
[2] AGH Akademia Górniczo-Hutnicza w Krakowie
[3] Katedra Teleinformatyki, Politechnika Gdańska

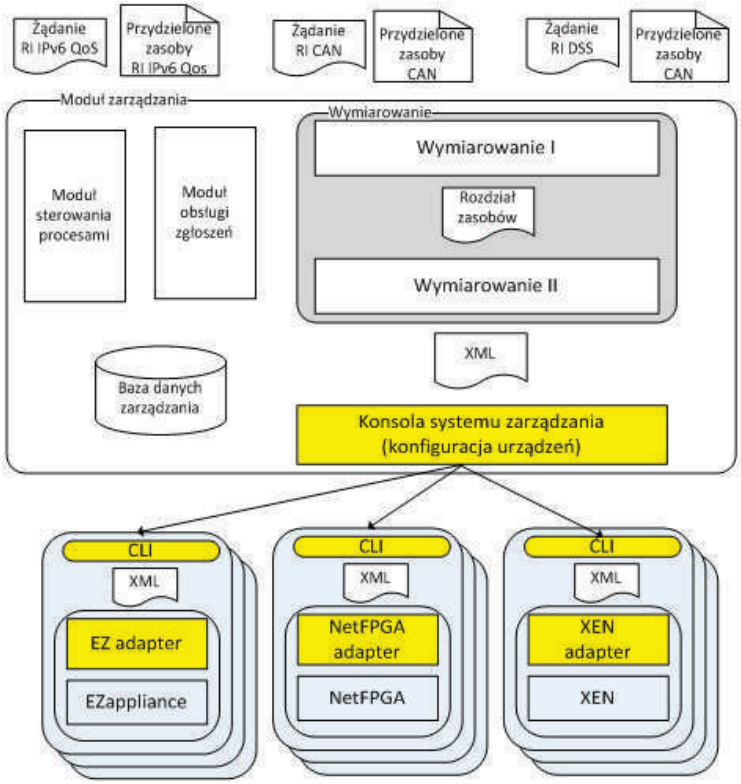
**Streszczenie** Celem artykułu jest przedstawienie współpracy dwuetapowego wymiarowania z systemem zarządzania na poziomie warstw L1 i L2 architektury Systemu IIP. Omówiono zadania systemu zarządzania oraz modułu wymiarowania, zrealizowane algorytmy optymalizacyjne oraz uniwersalne styki z systemem zarządzania oraz Równoległymi Internetami.

## 1 Wprowadzenie

Architektura Systemu IIP [1]-[4] realizowanego w ramach projektu Inżynieria Internetu Przyszłości (IIP) składa się z 4 podstawowych warstw: fizycznej (L1), wirtualizacji (L2), Równoległych Internetów (RI) (L3) oraz sieci wirtualnych (L4). Dwie pierwsze warstwy (L1/L2) stanowią infrastrukturę dla stworzenia tzw. Równoległych Internetów w warstwie trzeciej (L3). Koncepcja Równoległych Internetów [1] polega na możliwości działania niezależnych sieci, które mogą mieć różne protokoły dla płaszczyzny danych i sterowania, a jednocześnie współdzielą tę samą infrastrukturę fizyczną sieci dzięki technice wirtualizacji węzłów i łączy. W ramach projektu IIP trwają prace nad definicją trzech Równoległych Internetów: IPv6 QoS (*Quality of Service*), CAN (*Content Aware Networks*) i DSS (*Data Streams Switching*). Bardzo ważną rolę w Systemie IIP spełnia system zarządzania, którego istotnym elementem jest wymiarowanie mające za zadanie rozdzielenie zasobów elementarnych pomiędzy poszczególne RI. Zawartość niniejszego artykułu jest następująca: w rozdziale 2 przedstawiono podstawową funkcjonalność systemu zarządzania dla Systemu IIP. Rozdział 3 opisuje wymiarowanie zasobów z podziałem na dwa etapy: *Wymiarowanie I*, gdzie następuje rozdział zasobów fizycznych przy użyciu algorytmów optymalizacyjnych oraz *Wymiarowanie II* odpowiedzialne za implementację uzyskanych wyników w urządzeniach sieciowych W rozdziale 4 podsumowano obecny stan implementacji tego modułu oraz określono dalsze plany.

## 2 Podstawowa funkcjonalność systemu zarządzania

System zarządzania dla Systemu IIP wspomaga następujące procesy: (1) obsługę żądania utworzenia Równoległych Internetów, (2) przydział zasobów na podstawie przyjętego mechanizmu wymiarowania, (3) monitorowanie wykorzystania zasobów oraz (4) zwolnienie zasobów Równoległego Internetu. System zarządzania działa jak mediator pomiędzy RI a Systemem IIP i stąd jego rola jest niezmiernie istotna dla poprawnego działania całego Systemu IIP. Podstawowy system zarządzania zaimplementowany w projekcie IIP przedstawiono na rysunku 1. W wersji podstawowej moduł zarządzania składa się z modułu sterowania procesami, modułu wymiarowania, bazy danych i konsoli. Integralną częścią systemu zarządzania są adaptery w urządzeniach umożliwiających wirtualizację, które umożliwiają ustawienie parametrów tych urządzeń.




**Rysunek 1.** Podstawowy system zarządzania w projekcie IIP

*Moduł sterowania procesami* steruje uruchamianiem procesów odpowiedzialnych za zarządzanie. *Baza danych zarządzania* zawiera wszystkie niezbędne informacje dotyczące topologii, profili ruchu i szczegółowych wymagań od poszcze-

gólnych RI. *Moduł obsługi zgłoszeń* odpowiada za przyjmowanie żądań utworzenia RI oraz dostarczania informacji zwrotnej o przydzielonych zasobach. Moduł wymiarowania (*Wymiarowanie*) odpowiada za przydzielenie zasobów zgodnych z żądaniem utworzenia RI. *Konsola systemu zarządzania* pozwala na połączenie się z urządzeniami umożliwiającymi wirtualizację i ustawienie parametrów przydziału zasobów. W wersji podstawowej wykorzystywany jest interfejs typu CLI (*Command Line Interface*). Oprogramowanie urządzeń umożliwiających wirtualizację zostało rozszerzone o dedykowany *adapter*, który pozwala na konfigurowanie parametrów urządzeń z poziomu systemu zarządzania.

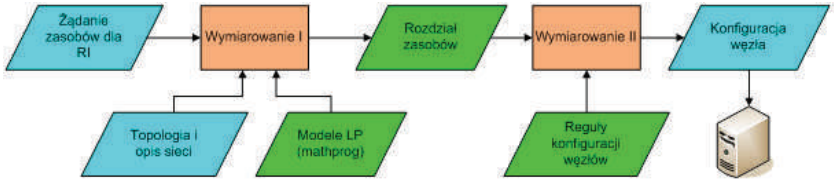
## 3 Dwupoziomowe wymiarowanie zasobów

Niniejszy rozdział zawiera opis modułu przydziału zasobów (wymiarowania) w sieci realizującego proces tworzenia wirtualnych węzłów i łączy dla poszczególnych RI. Zasoby sieci podlegające wymiarowaniu obejmują: przepustowości łączy, pamięć RAM, pamięć masową, moc obliczeniową w węzłach systemu oraz bufory (kolejki). Moduł wymiarowania dysponuje informacją o parametrach sieci dostępną w bazie danych zarządzania i realizuje optymalizację przydziału zasobów.

Proces wymiarowania definiuje topologię każdego RI. Na podstawie informacji o dostępnych zasobach, lokalizacji węzłów oraz wymagań odnośnie wydajności sieci, proces wymiarowania określa, które węzły i łącza wirtualne będą stanowiły topologię wymiarowanego RI oraz jaką część zasobów węzła i łącza należy przypisać dla danego RI.

Proces wymiarowania jest rozdzielony na dwa etapy (Rys. 2):

- *Wymiarowanie I* – odpowiada za rozdział dostępnych zasobów dla poszczególnych RI na podstawie wybranego modelu optymalizacyjnego lub heurystyki.
- *Wymiarowanie II* – odpowiada za tłumaczenie rozwiązania ogólnego sformułowanego przez *Wymiarowanie I* na ustawienia platform sprzętowych w węzłach sieci.




**Rysunek 2.** Relacja pomiędzy systemem zarządzania a procesem wymiarowania ze wskazaniem wymienianych danych

Proces przypisania zasobów decyduje o przydziale konkretnych zasobów do RI. Stanowi on łącznik między wysokopoziomowym opisem sieci uzyskanym

w procesie wymiarowania a niskopoziomowym opisem dostępnych urządzeń. W praktyce proces ten generuje pliki z opisem wirtualnych węzłów, a także aktualizuje bazę danych zawierającą informacje o zasobach.

Proces konfiguracji węzłów odpowiada za utworzenie węzła wirtualnego w środowisku wirtualizacyjnym. Na podstawie pliku z opisem węzła proces ten konfiguruje zarządcę wirtualizacji i uruchamia maszyny wirtualne.

### 3.1 *Wymiarowanie I*

Na obecnym etapie rozwoju systemu trwają prace nad szczegółowymi algorytmami wymiarowania, ich zakresem stosowalności oraz dostosowaniem do potrzeb poszczególnych RI.

Zaimplementowano 3 modele optymalizacyjne, których celem jest rozdział przepustowości łączy oraz mocy obliczeniowej dostępnej w węzłach na podstawie żądań zdefiniowanych przez poszczególne RI. Szczegóły modeli można znaleźć w [5]. Obecnie uwzględnione funkcje celu obejmują następujące przypadki:

1. Minimalizację zajętej przepustowości całkowitej:

$$\textbf{minimize} \quad F = \sum_i \sum_e \sum_v x_{iev} \tag{1}$$

   gdzie $x_{iev} > 0$ oznacza przepustowość zaalokowaną dla $i$-tego RI w łączu $e$ incydentnym w stosunku do węzła $v$ (wyrażoną w Mbit/s).

2. Minimalizację użycia zasobów węzłowych:

$$\textbf{minimize} \quad F = \sum_i \sum_v y_{iv} \tag{2}$$

   gdzie $y_{iv} > 0$ oznacza oznacza moc obliczeniową dostępną dla $i$-tego RI w węźle $v$ (wyrażoną w Mflopach).

3. Minimalizację zajętej przepustowości (analogicznie do sytuacji 1) i dodatkowo spełnienie wymagań na całkowite opóźnienie transmisyjne na ścieżce dla żądania $d$ z $i$-tego RI mniejsze niż określone jako $g_{id}$:

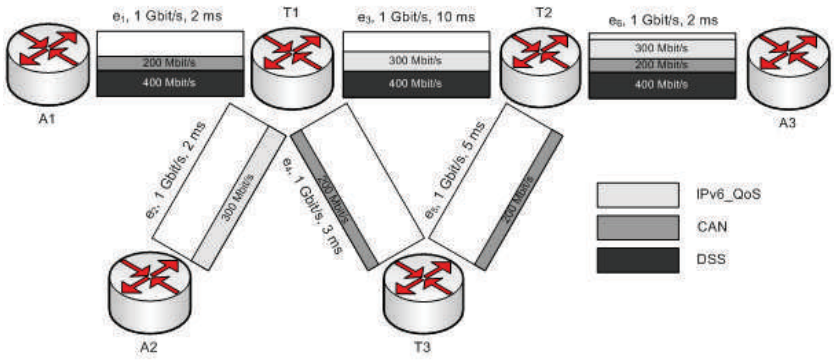
$$\sum_e n_{ied} f_e \leqslant g_{id} \quad i \in I \quad d \in D \tag{3}$$

   gdzie $f_e$ oznacza maksymalne opóźnienie transmisyjne wzdłuż łącza $e$, natomiast $n_{ied}$ przyjmuje wartość 1 lub 0 wskazując odpowiednio czy żądanie $d$ $i$-tego RI realizowane jest przez łącze $e$ czy też nie.

Główne problemy do rozwiązania w tym obszarze obejmują definicję pojemności sieci, definicję pojemności węzła sieci, rozpoznanie zależności między pojemnością węzła fizycznego a pojemnością węzłów wirtualnych, jak również stworzenie metodologii dla wymiarowania sieci, która uwzględniałaby zależność między wymaganą pojemnością łącza wirtualnego a pojemnością węzła wirtualnego.

Podstawowe sformułowanie problemu wymiarowania obejmuje identyfikację potrzeb poszczególnych RI, rozpoznanie zasobów dostępnych w węzłach oraz w łączach, a następnie przydzielenie zasobów ze sprawdzeniem możliwości realizacji przydziału zasobów. Kryterium inicjowania procedury wymiarowania jest generowane przez system zarządzania.

W celu zilustrowania efektów działania algorytmów zaimplementowanych w module *Wymiarowanie I* na rysunku 3 przedstawiono prosty przykład podziału przepustowości łączy pomiędzy trzy RI, tj. IPv6_QoS, DSS oraz CAN.




**Rysunek 3.** Ilustracja podziału zasobów transportowych (przepustowości) po pierwszym etapie wymiarowania w sieci o uproszczonej architekturze

Węzły z etykietami $A_1 \div A_3$ oznaczają węzły dostępowe, natomiast węzły tranzytowe oznaczone są etykietami $T_1 \div T_3$. Żądania przepustowości zdefiniowane pomiędzy węzłami dostępowymi dla poszczególnych RI mają następujące wartości: DSS – 400 Mbit/s pomiędzy $A_1$ i $A_3$, CAN – 200 Mbit/s pomiędzy $A_1$ i $A_3$ oraz IPv6_QoS – 300 Mbit/s pomiędzy $A_2$ i $A_3$. Na łączach zaznaczono ich etykiety ($e_1 \div e_6$), dostępną przepustowość (taką samą dla wszystkich łączy i równą 1 Gbit/s) oraz opóźnienie transmisyjne na poszczególnych łączach. Wyniki przedstawiono graficznie dla modelu optymalizacyjnego, którego zadaniem jest minimalizacja wykorzystanej przepustowości oraz spełnienie wymagań dotyczących maksymalnego dopuszczalnego opóźnienia transmisyjnego. W przedstawionym przypadku ograniczenia dotyczące opóźnień transmisyjnych ustawiono następująco: DSS – 15 ms, CAN – 12 ms oraz IPv6_QoS – 20 ms. Efektem działania zaproponowanych algorytmów są topologie RI z łączami o przydzielonych przepustowościach. W prezentowanym przykładzie minimalizacja przydzielonej przepustowości oznacza realizację żądań po najkrótszych ścieżkach. Topologia RI DSS składa się z węzłów $A_1$, $T_1$, $T_2$ i $T_3$ oraz łączy $e_1$, $e_3$ i $e_6$ o przydzielonej przepustowości 400 Mbit/s na każdym z nich. Podobna topologia została wyznaczona dla RI IPv6_QoS, z tym, że zamiast węzła $A_2$ i łącza $e_1$ użyto węzła $A_2$ i łącza $e_2$. Przydzielona przepustowość na łączach $e_2$, $e_3$ i $e_6$ jest równa 300 Mbit/s. Odmienna topologia została wyznaczona dla RI CAN. Z powodu

ograniczenia na opóźnienie transmisyjne jedynie topologia złożona z węzłów $A_1$, $T_1$, $T_3$, $T_2$ i $A_3$ oraz łączy $e_1$, $e_4$, $e_5$ i $e_6$ spełnia nałożone wymagania i zostaje zaakceptowana jako wynik działania modułu *Wymiarowanie I* oraz przekazana do modułu *Wymiarowanie II* w celu translacji ustawień oraz realizacji wpisu do platform implementacyjnych.

### 3.2 *Wymiarowanie II*

Zadaniem modułu *Wymiarowanie II* jest odwzorowanie wyników *Wymiarowania I* na konkretne parametry platform implementacyjnych. W opisywanym problemie uzyskujemy 3 stopnie swobody i konieczność uwzględnienia:

- 3 platform implementacyjnych (Xen, NetFPGA oraz EZAppliance),
- 3 typów RI,
- specyfiki aplikacji i charakteru strumieni ruchu na obsługę w węzłach, dokładne ustawienia *schedulera*, zapotrzebowanie na zasoby węzłowe, np. moc obliczeniową czy pamięć RAM.

Dla platformy XEN i netFPGA należy ustawić: *CycleLength* (długość cyklu), *NumberPhases* (liczbę faz), *PhaseLength* (długość fazy) odpowiednio dla RI IPv6_QoS, RI CAN i RI DSS. Natomiast platforma EZAppliance wymaga ustawienia: *BaseValue*, *Weight*, CIR (*Committed Information Rate*) i CBS (*Committed Burst Size*). Dla platformy XEN niezbędny jest dodatkowy algorytm korekcyjny pozwalający na ustawianie zadanej przepływności. Niewielkie korekcje wynikają z technicznych ograniczeń realizacji *schedulera*. Tabela 1 zawiera przykładowe wartości parametrów łącza, które uzyskujemy z *Wymiarowania I* i parametrów specyficznych dla platformy EZAppliance, które pozwalają na ustawienie zadanych parametrów łącza.

**Tablica 1.** Przykładowe wartości parametrów dla platformy EZAppliance

| Parametry łącza | Parametry IPv6_QoS | Parametry CAN | Parametry DSS |
|---|---|---|---|
| IPv6_QoS = 400 Mbit/s<br>CAN = 100 Mbit/s<br>DSS = 500 Mbit/s | *Weight* = 4<br>CIR = 50000000<br>CBS = 1500 | *Weight* = 4<br>CIR = 12500000<br>CBS = 1500 | *Weight* = 4<br>CIR = 62800000<br>CBS = 1500 |
| IPv6_QoS = 400 Mbit/s<br>CAN = 200 Mbit/s<br>DSS = 400 Mbit/s | *Weight* = 4<br>CIR = 50000000<br>CBS = 1500 | *Weight* = 2<br>CIR = 25000000<br>CBS = 1500 | *Weight* = 2<br>CIR = 25000000<br>CBS = 1500 |

Na rysunku 4 została przedstawiona konfiguracja interfejsu dla łącza $e_3$ w węźle $T_1$ w wersji dla platformy EZappliance. W tym przypadku ustawienia parametrów mają specyficzny charakter związany z architekturą sprzętową tej platformy i nawiązują do wskazówek znajdujących się w dokumentacji i są również zgodne z obserwacjami wyników testów.

```
<Node id="T1">
<Interfaces>
  <Interface id="eth3">
    <Scheduler type="EZ">
      <BaseValue>1000</BaseValue>
      <Weight pi="QOS">1</Weight>
      <CIR pi="QOS">37500000</CIR>
      <CBS pi="QOS">1500</CBS>
      <Weight pi="DSS">1</Weight>
      <CIR pi="DSS">50000000</CIR>
      <CBS pi="DSS">1500</CBS>
    </Scheduler>
  </Interface>
</Interfaces>
</Node>
```

**Rysunek 4.** Konfiguracja sprzętowa interfejsu dla łącza $e_3$ w węźle $T_1$ dla platformy EZappliance dla parametrów przedstawionych na rysunku 3

Na rysunku 5 z kolei przedstawiono konfigurację interfejsu dla łącza $e_6$ w węźle $T_2$ w wersji dla platformy Xen. Podobnie jak w przypadku poprzednim, ustawienia parametrów mają specyficzny charakter związany z architekturą sprzętową tej platformy. Cykl 100000 $\mu$s na łączu 1 Gbit/s został podzielony proporcjonalnie do zaalokowanych zasobów w pierwszym etapie wymiarowania dla trzech RI: IPv6_QoS (QOS), CAN i DSS. Niewykorzystana przepustowość została oznaczona jako faza o identyfikatorze ″NIL″.

```
<Node id="T2">
  <Interfaces>
    <Interface id="eth1">
      <Scheduler type="XenNet">
        <Cyclelength>100000</Cyclelength>
        <NumberPhases>4</NumberPhases>
        <PhaseLength pi="QOS">30000</PhaseLength>
        <PhaseLength pi="CAN">20000</PhaseLength>
        <PhaseLength pi="DSS">40000</PhaseLength>
        <PhaseLength pi="NIL">10000</PhaseLength>
      </Scheduler>
    </Interface>
  </Interfaces>
</Node>
```

**Rysunek 5.** Konfiguracja sprzętowa łącza $e_6$ w węźle $T_2$ dla architektury Xen dla parametrów przedstawionych na rysunku 3

## 4 Podsumowanie

Przy specyfikowaniu architektury i funkcjonalności systemu zarządzania dla Systemu IIP, w celu zapewnienia elastyczności i efektywności działania systemu w warstwach L1/L2, zdefiniowano moduł wymiarowania, który zapewnia rozdział zasobów pomiędzy poszczególne RI z uwzględnieniem zmian zapotrzebowania oraz sytuacji awaryjnych. Zostały przeprowadzone testy pierwszej wersji implementacji zarówno algorytmów wysokopoziomowego rozdziału zasobów sieciowych i węzłowych, jak również bezpośrednią konfigurację parametrów platform sprzętowych.

## Literatura

1. W. Burakowski, H. Tarasiuk, A. Bęben, *Architektura Systemu IIP*, Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź, 14-16 września 2011.
2. W. Burakowski et al., *The IIP System: Architecture, Parallel Internets, Virtualization and Applications*, Future Network & Mobile Summit 2011, Warszawa, 15-17 czerwca 2011.
3. W. Burakowski, H. Tarasiuk, A. Bęben, *Future Internet Architecture Based on Virtualization and Co-existence of Different Data and Control Planes*, 5*th* Future Internet Cluster Workshop focused on Future Network Architectures, Warszawa, 14 czerwca 2011.
4. W. Burakowski, H. Tarasiuk, A. Bęben, *System IIP for supporting "Parallel Internets (Networks)"*, Future Internet Assembly Meeting, Ghent, 2010.
5. J. Gozdecki, M. Kantor, K. Wajda, J. Rak, *A Flexible Provisioning Module for Optimizing Utilization of Resources for the Future Internet IIP Initiative*, Networks 2012, 15*th* International Telecommunications Strategy and Planning Symposium, Rome, 15-18 October, 2012.

# Implementacja i testy architektury bezpieczeństwa na poziomie 2 Systemu IIP

Jerzy Konorski[1], Krzysztof Cabaj[2], Zbigniew Kotulski[3], Paweł Szałachowski[3], Grzegorz Kołaczek[4], Jerzy Kasperek[5], Piotr Pacyna[5], Dominik Rzepka[5], Wojciech Romaszkan[5], Marcin Rupiński[5], Jehoszafat Zimnowoda[5], Andrzej Kamisiński[5], Paweł Rajda[5]

[1] Wydział Elektroniki, Telekomunikacji i Informatyki, Politechnika Gdańska
[2] Instytut Informatyki, Politechnika Warszawska
[3] Instytut Telekomunikacji, Politechnika Warszawska
[4] Instytut Informatyki, Politechnika Wrocławska
[5] AGH Akademia Górniczo-Hutnicza w Krakowie

**Streszczenie** Projekt Inżynieria Internetu Przyszłości (IIP) rozwija koncepcję infrastruktury transmisyjnej, stanowiącej jednolity system komunikacyjny (System IIP) do obsługi strumieni danych pochodzących od trzech rodzajów sieci nazywanych Równoległymi Internetami. Architektura tego systemu obejmuje cztery poziomy, przy czym poziom 2 odpowiada za tworzenie i utrzymywanie łączy i węzłów wirtualnych. Niniejsze opracowanie przedstawia rozszerzony opis trzech linii obrony stanowiących architekturę bezpieczeństwa dla tego systemu na poziomie 2, która została opracowana w celu przeciwdziałania atakom opartym na wprowadzaniu obcego ruchu sieciowego do Systemu IIP, a także w celu przeciwdziałania manipulacji bądź fałszowaniu ruchu użytkowego i sygnalizacyjnego Systemu IIP. Przedstawiono wczesne doświadczenia z implementacji mechanizmów obronnych oraz omówiono wyniki testów przeprowadzonych w środowisku PL-LAB. Wyniki przedstawione w tym rozdziale są rozszerzoną i uaktualnioną wersją rezultatów zaprezentowanych w artykułach [1] i [2].

## 1 Wprowadzenie

Projekt Inżynieria Internetu Przyszłości (IIP) [3] rozwija koncepcję infrastruktury transmisyjnej oraz sieci nazywanych Równoległymi Internetami, tworzących łącznie jednolity system komunikacyjny Internetu Przyszłości. Infrastruktura transmisyjna System IIP jest przeznaczona do przesyłania jednostek danych IIP-PDU, nazywanych ramkami IIP. Każdy Równoległy Internet wykorzystuje łącza i węzły wirtualne wydzielone w oparciu o fizyczne łącza i przełączniki oraz definiuje swój własny stos protokolarny. Dwa spośród Równoległych Internetów: *Data Stream Switching* (DSS) i *Content Aware Network* (CAN) są rozwiązaniami określanymi jako post-IP, zaś trzeci jest oparty na protokole IPv6 ze wsparciem jakości usług (IPv6 QoS). Istnieje także czwarty rodzaj Internetu, *Management*, także oparty na protokole IPv6, będący wydzieloną siecią zarządzania Systemem IIP.

Architektura Systemu IIP zawiera cztery poziomy, przy czym poziom 2 odpowiada za tworzenie i utrzymywanie łączy i węzłów wirtualnych.

W celu przeciwdziałania zagrożeniom na tym poziomie zaproponowano mechanizmy bezpieczeństwa koncentrujące się na:

1. zagrożeniach zewnętrznych (intruzach) dokonujących manipulacji ruchu sieciowego, bądź wprowadzających nielegalny ruch do Systemu IIP, by zakłócić jego funkcjonowanie, jak też
2. wewnętrznych węzłach wirtualnych przejętych przez intruzów w celu fałszowania legalnego ruchu Systemu IIP dla osiągnięcia różnych szkodliwych celów.

Przyjmuje się zatem za możliwe do przeprowadzenia ataki metodą wprowadzania obcego ruchu (*traffic injection*), metodą powtarzania bądź zaburzania kolejności IIP-PDU (*replay/resequencing/reordering*), metodą zaburzania relacji czasowych w strumieniach IIP-PDU, np. poprzez przechwytywanie i przetrzymywanie ramek (*ruffling*) oraz metodą generacji fałszywego ruchu Systemu IIP (*forging*). W pracach [4] i [5] (por. też [6]) zaproponowano architekturę bezpieczeństwa dla poziomu 2 Systemu IIP, składającą się z trzech linii obrony. Pierwsza linia obrony odpowiada za zabezpieczenie ramek za pomocą kryptograficznej sumy kontrolnej HMAC oraz za obsługę numerów sekwencyjnych jednostek IIP-PDU dla zapobiegania atakom metodami *traffic injection* oraz *replay/resequencing/reordering*. Weryfikację HMAC przeprowadza się w węźle wirtualnym po odbiorze każdej ramki. Druga linia obrony wykrywa anomalne zachowania strumienia ruchu bądź węzła spowodowane atakami z użyciem metod zaburzania relacji czasowych lub generacji fałszywego ruchu. W tym celu obserwuje się zdarzenia związane z zagrożeniami bezpieczeństwa (*Security-Relevant Events*, SRE). SRE analizowane są przez umieszczone w węzłach transmisyjnych moduły wykrywania anomalii (*Local Anomaly Detection*, LAD), działające w ramach szerszej jednostki funkcjonalnej pod nazwą lokalny agent bezpieczeństwa (*Local Security Agent*, LSA). Anomaliom noszącym znamiona ataku przypisywane są miary liczbowe, które pozwalają przekształcić je w lokalne metryki reputacji węzła. Metryki te są przekazywane do trzeciej linii obrony, gdzie właściwy dla danego Równoległego Internetu główny agent bezpieczeństwa (*Master Security Agent*, MSA) wypracowuje na ich podstawie miary zaufania związane z każdym węzłem tego Internetu. Ponadto, wykorzystując moduł wykrywania anomalii o zasięgu całego Równoległego Internetu (*Parallel Internet-wide Anomaly Detection*, PI-AD) MSA identyfikuje także anomalie o zasięgu szerszym niż zasięg pojedynczego węzła.

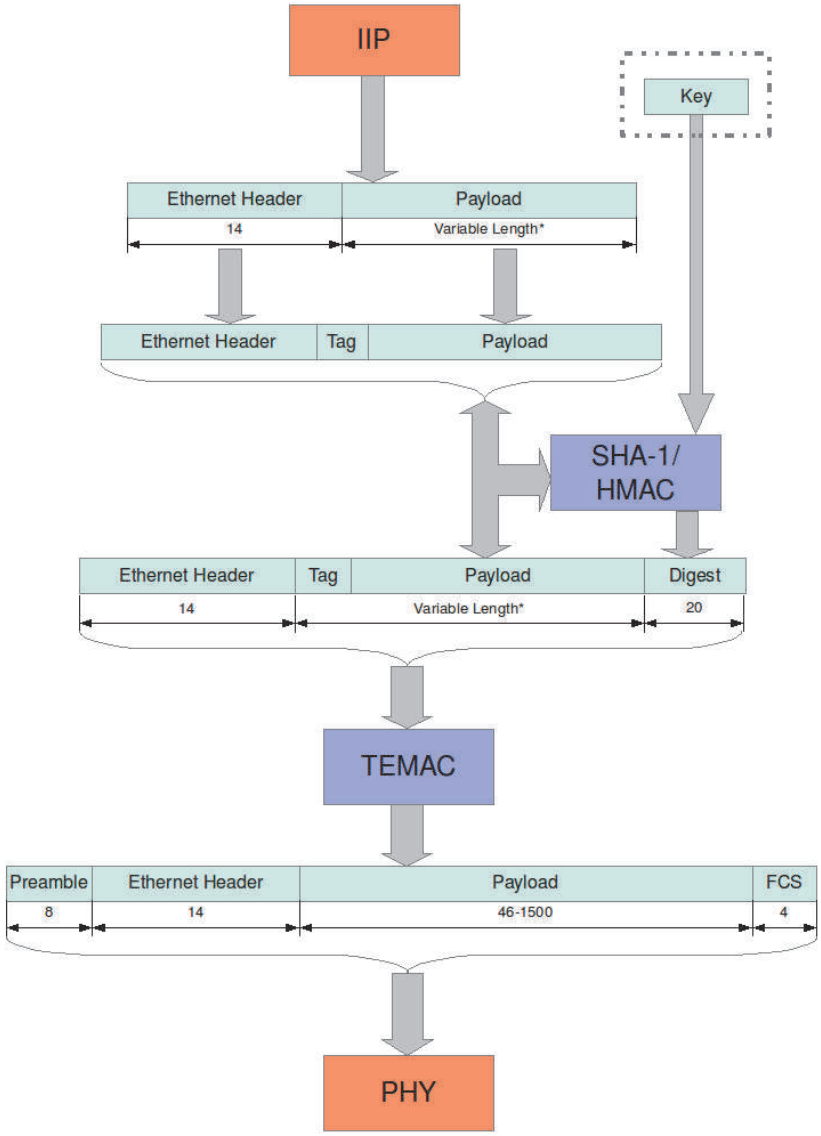
Niniejszy rozdział jest oparty na wczesnych doświadczeniach z integracji wybranych mechanizmów bezpieczeństwa z siecią zarządzania Systemu IIP przeprowadzonej w środowisku PL-LAB. Przedstawiamy wyniki uzyskane w zakresie implementacji i testowania działania wszystkich trzech linii obrony.

## 2 Implementacja i testowanie modułów HMAC-SHA-1

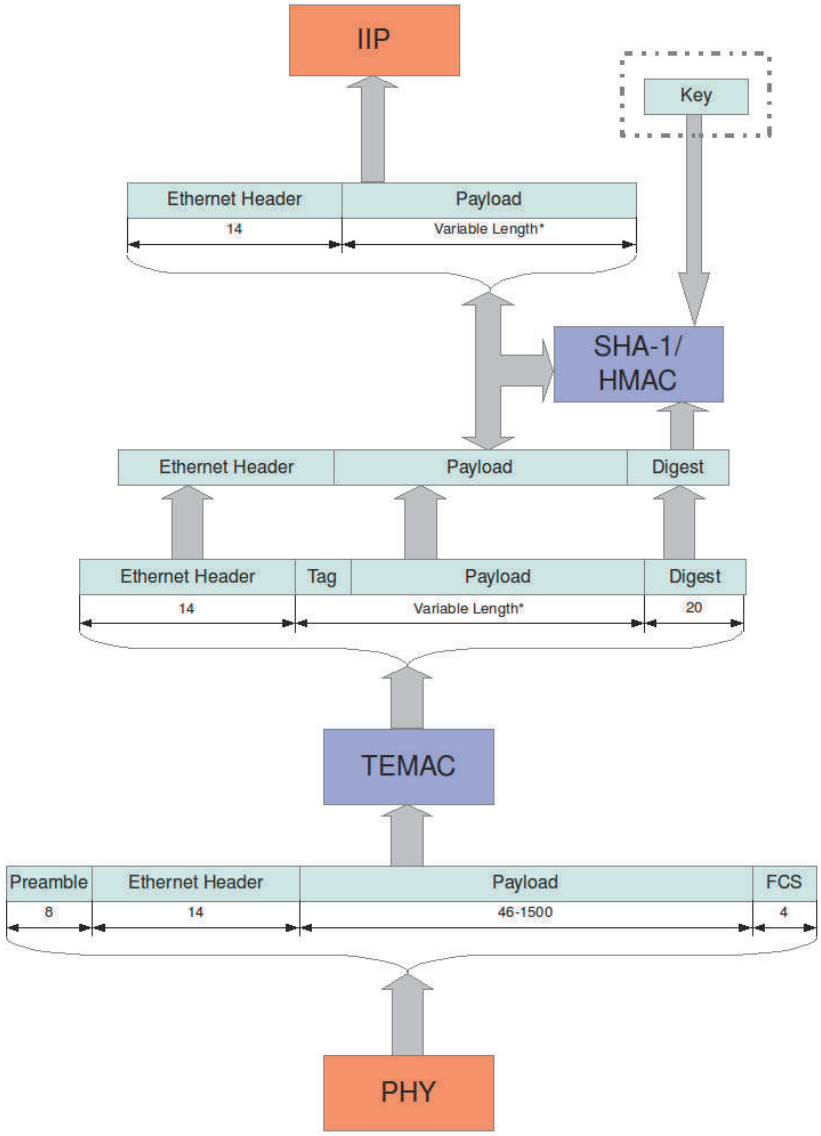
Sprzętowy koder oraz weryfikator HMAC-SHA-1 zapewnia jednolitą ochronę zagregowanego strumienia danych, który powstaje z multipleksacji strumieni danych przesyłanych przez trzy rodzaje sieci IPv6-QoS, CAN i DSS w systemie transmisyjnym IIP. Na poziomie warstwy łącza danych ochronie podlegają ramki IIP, zawarte w strukturze ramki Ethernet. Zaletą omawianego rozwiązania jest unifikująca forma ochrony strumienia danych w systemie. Jest niezależna od rodzaju aplikacji, od których pochodzą przenoszone dane, niezależna od specyfiki Równoległego Internetu oraz niezależna od stosowanych protokołów wyższych warstw. Implementacja zapewnia ochronę integralności ramki Ethernet wraz z nagłówkiem, w tym z polami nadawcy i odbiorcy. W powiązaniu z wprowadzoną w tym rozwiązaniu obsługą numerów sekwencyjnych omawiana implementacja umożliwia wykrywanie niektórych form ataków przeciwko strumieniowi danych, takich jak na przykład wprowadzanie obcego ruchu, fałszowanie przesyłanych danych oraz atak poprzez powtórzenia (*replay*). Przetwarzanie ramek przez moduł HMAC-SHA-1 polega na wytworzeniu sygnatury skrótu HMAC-SHA-1 i numeru sekwencyjnego ramki oraz na dołączeniu tych informacji do ramki po stronie nadawcy, a potem na weryfikacji poprawności tej ramki poprzez ponowne obliczenie sygnatury po stronie odbiorcy i porównanie jej z sygnaturą otrzymaną, a także sprawdzeniu numeru sekwencyjnego, co skrótowo ilustrują rysunki 1 i 2.

### 2.1 Architektura sprzętowa HMAC-SHA-1

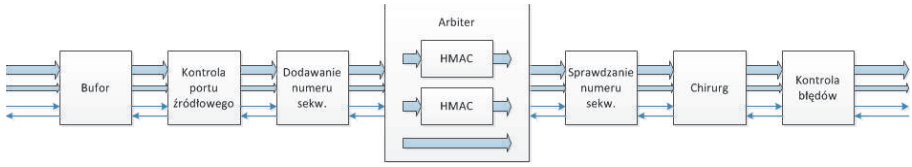
Warstwa sprzętowa odpowiedzialna za przetwarzanie danych została zaimplementowana w postaci rekonfigurowalnej logiki wewnątrz układu Virtex-II Pro znajdującego się na karcie NetFPGA 1G. Składa się ona z modułów napisanych w języku opisu sprzętu Verilog, połączonych szeregowo. Każda ramka Ethernet przesyłana wewnątrz struktury NetFPGA jest opatrzona co najmniej jednym nagłówkiem sterującym, zawierającym informacje niezbędne do prawidłowego przetwarzania tej ramki. Ramki przychodzące z czterech zewnętrznych portów fizycznych o przepustowości nominalnej 1 Gbit/s każdy oraz z czterech portów wirtualnych, dostępnych od strony systemu operacyjnego Linux przez magistralę PCI trafiają do tak zwanej ścieżki danych w układzie FPGA, gdzie znajdują się moduły odpowiedzialne za przetwarzanie ich zawartości. Każdy moduł może być dowolnie modyfikowany lub podmieniany, zależnie od potrzeb projektanta. Spójność całej struktury zapewnia opracowany na Uniwersytecie Stanforda standard komunikacji między modułami wewnątrz struktury NetFPGA. Dane są przesyłane wzdłuż ścieżki danych w 64-bitowych blokach razem z 8-bitowym polem sterującym. Pozwala to na realizację algorytmów o przetwarzaniu potokowym. Aby ułatwić komunikację z oprogramowaniem, karta NetFPGA posiada własny system rejestrów dostępnych z poziomu systemu operacyjnego za pomocą funkcji *ioctl()*. Rejestry definiowane są w plikach *.xml* i generowane za pomocą skryptów w języku Perl. Projektant może dodawać lub modyfikować rejestry na potrzeby




**Rysunek 1.** Przetwarzanie sygnatury HMAC-SHA-1: generacja




**Rysunek 2.** Przetwarzanie sygnatury HMAC-SHA-1: weryfikacja




**Rysunek 3.** Modularna struktura HMAC-SHA-1

swojej aplikacji. Moduł HMAC-SHA-1 wprowadza zarówno rejestry użytkownika jak również wewnętrzne rejestry techniczne.

## 2.2 Architektura wewnętrzna rozwiązania HMAC-SHA-1

HMAC-SHA-1 składa się z modułów realizujących odrębne operacje na ramce (rysunek 3).

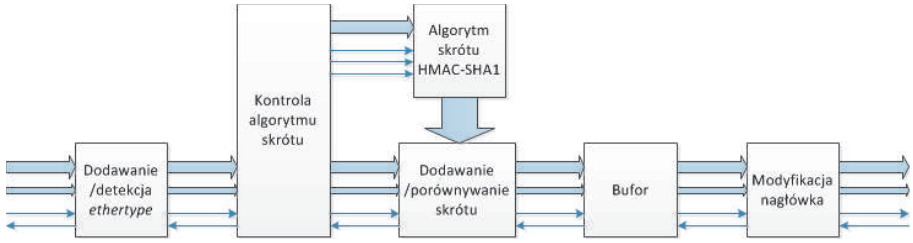
**Bufor** o pojemności 16kB, zrealizowany z użyciem wbudowanych pamięci BRAM, zapobiega utracie ramek w sytuacji, gdy chwilowy napływ ruchu przekracza możliwości przetwarzania przez modułu HMAC-SHA-1.

**Kontrola portu źródłowego** porównuje rodzaj ramki (zabezpieczona albo niezabezpieczona) z konfiguracją portu, przez który została odebrana (port do odbioru ruchu zabezpieczonego albo niezabezpieczonego). Przedmiotem uwagi jest wartość pola *ethertype*. Wartość umowna *custom ethertype* wskazuje, że ramka jest zabezpieczona. Ze względu na to, że na nagłówek sterujący ramki w układzie NetFPGA przeznaczone jest 8 bajtów, a jego standardowe pola informacyjne wypełniają 5 bajtów, pozostała przestrzeń jest wykorzystywana wewnątrz HMAC-SHA-1 do przekazywania dodatkowych informacji o stanie przetworzenia ramki w miarę jej przekazywania pomiędzy modułami. Ramki niewłaściwego typu są znakowane w dodatkowym polu tego nagłówka i ostatecznie wysyłane na port diagnostyczny do analizy. Numer portu diagnostycznego, podobnie jak inne parametry układu są definiowane w pliku konfiguracyjnym.

**Dodawanie numeru sekwencyjnego** przydziela ramce odpowiedni numer bezpiecznego powiązania (asocjacji) oraz numer sekwencyjny w obrębie tej asocjacji na podstawie wartości licznika ramek.

**Arbiter** dzieli przychodzący ruch pomiędzy dwa moduły HMAC-SHA-1 pracujące równolegle, a następne scala wychodzący z nich ruch w jeden strumień danych, z zachowaniem właściwej kolejności ramek. Ramki przeznaczone przez moduł kontroli portu źródłowego do odrzucenia omijają przetwarzanie w module HMAC i są kierowane bezpośrednio do wyjścia z modułu arbitra.

**Moduł HMAC** (rysunek 4) realizuje obliczanie wartości skrótu za pomocą algorytmu HMAC-SHA-1 oraz dodawanie skrótu do ramki albo weryfikację wartości skrótu zawartego w ramce. Moduł dodawania albo detekcji pola custom ethertype decyduje o trybie działania modułu – w przypadku wykrycia obecności charakterystycznego dla Systemu IIP nagłówka ustawiany jest tryb weryfikacji ramki. W przeciwnym wypadku aktywowany jest tryb zabezpieczania




**Rysunek 4.** Struktura wewnętrzna modułu HMAC

i jest dodawane pole *custom ethertype*. Sprawdzany jest również numer asocjacji przypisany do ramki i na jego podstawie ładowane są stosowne klucze kryptograficzne dla obliczenia skrótu przy użyciu algorytmu SHA-1. Moduł kontrolujący algorytm obliczania skrótu wydziela część ramki przeznaczoną do przetworzenia. Algorytm skrótu HMAC-SHA1 oblicza wartość skrótu SHA-1 z danych wejściowych połączonych ze stosownym kluczem. Wynikowy skrót jest ponownie łączony z kluczem i przetwarzany przez algorytm SHA-1. W zależności od trybu (*protect/verify*), obliczony skrót jest doklejany do niezabezpieczonej ramki, bądź porównywany ze skrótem zawartym w zabezpieczonej ramce. Jeżeli moduł HMAC pracuje w trybie weryfikacji ramek zabezpieczonych, to po wykryciu niezgodności skrótów: obliczonego i otrzymanego w ramce, musi zmodyfikować nagłówek ramki tak, aby została ona wysłana na interfejs diagnostyczny do analizy. Ponieważ skrót jest zawarty w końcowej części ramki, a informacja o jego poprawności musi zostać zapisana w nagłówku ramki, to na czas weryfikacji skrótu ramka jest buforowana. Po stwierdzeniu poprawności ramki modyfikowany jest odpowiedni bit w nagłówku ramki.

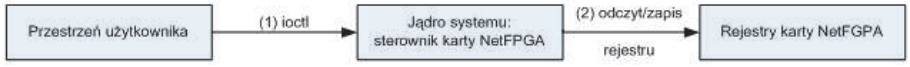
**Moduł sprawdzania poprawności numerów sekwencyjnych** realizuje algorytm krótko opisany niżej. W przypadku natrafienia na niepoprawny numer ramka zostaje oznaczona jako niepoprawna. Zapisany zostaje oczekiwany numer sekwencyjny, który następnie zostaje przekazany na interfejs diagnostyczny wraz z ramką.

**Moduł chirurga** realizuje usuwanie numeru sekwencyjnego, pola *custom ethertype* oraz skrótu z ramki zweryfikowanej jako poprawna.

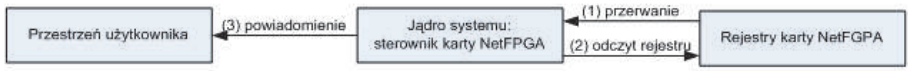
**Moduł kontroli błędów** analizuje pola błędów w nagłówku NetFPGA. Na ich podstawie zmienia wartości liczników rejestrujących statystyki dotyczące ramek. W przypadku ramki zakwalifikowanej jako błędna (z powodu błędnego numeru portu, niepoprawnej wartości skrótu lub błędu numeru sekwencyjnego) jej pierwsze 64 bajty są przekazywane na port diagnostyczny, wraz z dodatkowymi informacjami o typie błędu.

### 2.3 Oprogramowanie systemowe

**Komunikacja z systemem operacyjnym** Komunikacja ta obejmuje komunikację pomiędzy układem NetFPGA a sterownikiem karty w jądrze systemu ope-




**Rysunek 5.** Schemat komunikacji użytkownika systemu Linux z kartą NetFPGA: mechanizm *ioctl*




**Rysunek 6.** Schemat komunikacji użytkownika systemu Linux z kartą NetFPGA: mechanizm *Netlink*)

racyjnego Linux a także komunikację pomiędzy sterownikiem a aplikacją w przestrzeni użytkownika. Praca karty NetFPGA realizującej funkcję generatora lub weryfikatora HMAC-SHA-1 może być kontrolowana z poziomu systemu operacyjnego poprzez odczyt lub modyfikacje odpowiednich rejestrów urządzenia, z których część jest przynależna do karty netFPGA, zaś część jest specyficzna dla układu HMAC. Standardowo komunikacja ta odbywa się poprzez magistralę PCI. Rysunek 5 przedstawia zarządzanie rejestrami karty NetFPGA zrealizowaną poprzez systemowe funkcje *ioctl*.

Wadą tego rozwiązania jest brak możliwości komunikacji dwukierunkowej, bowiem inicjatorem komunikacji może być tylko proces z przestrzeni użytkownika w systemie operacyjnym. Od HMAC-SHA-1 wymaga się, aby również karta mogła powiadomić proces użytkownika o stanie w jakim się znalazła – na przykład o zmianach jakie zachodzą w jej rejestrach. W szczególności karta powinna mieć możliwość żądania zmiany aktywnej asocjacji (czyli tzw. bezpiecznego powiązania), a wraz z nim żądania aktywacji nowego klucza kryptograficznego w procesie wymiany klucza gdy bieżące bezpieczne powiązanie wygaśnie ze względu na upływ czasu, bądź z powodu wyczerpania zakresu dopuszczalnych numerów sekwencyjnych. Problem obsługi przerwań od karty rozwiązano rozszerzając sterownik karty NetFPGA w jądrze systemu Linux oraz wykorzystując kontroler przerwań karty.

W pełni dwukierunkową komunikację pomiędzy sterownikiem karty a procesem z przestrzeni użytkownika osiągnięto poprzez wprowadzenie gniazd sieciowych typu *Netlink* [7] - rysunek 6, dzięki któremu proces użytkownika ma możliwość nasłuchiwania zdarzeń pochodzących z jądra systemu, w tym zdarzeń zgłaszanych przez sterownik karty NetFPGA.

Przebieg komunikacji, w której inicjatorem jest karta NetFPGA, jest przedstawiony na rysunku 6. Kolejność zdarzeń jest następująca:

1. z karty NetFPGA do jej sterownika dociera przerwanie,

2. sterownik rozpoznaje typ przerwania poprzez odczyt odpowiedniego rejestru karty; następnie sterownik może odczytać który rejestr uległ zmianie oraz jaka jest aktualna wartość tego rejestru,
3. taki zestaw danych jest przekazywany do procesu z przestrzeni użytkownika, gdzie może zajść dalsza interpretacja danych oraz ich przetwarzanie.

Stosowanie gniazd typu *Netlink* pozwoliło na zdefiniowanie własnego protokołu komunikacji, który dodatkowo można wyposażyć w:

– mechanizmy potwierdzeń poprawnego zapisu danej wartości do danego rejestru karty NetFPGA,
– szczegółowy opis błędu, który wystąpił podczas wykonywania danej czynności przez sterownik karty,
– wiadomości typu multicast do wielu procesów działających w przestrzeni użytkownika.

Podstawowe problemy napotkane podczas implementacji tego schematu komunikacji pomiędzy sprzętem, sterownikiem a aplikacją związane były ze zmianami w mechanizmie *Netlink*, które od wersji jądra 2.6.28 ulegały znacznym modyfikacjom przy równoczesnym braku dokumentacji zmian oraz słabym wsparciu dla kolejnych wersji bibliotek w systemie CentOS. Ostatecznie wybrano nową wersję jądra systemu i zdecydowano się na użycie biblioteki *libnl-genl* będącej rozszerzoną wersją biblioteki *libnl*.

**Zarządzanie i sterowanie pracą układu** Układ jest w pełni zarządzany. Zapis i odczyt rejestrów może być realizowany lokalnie z wiersza poleceń poprzez interfejs *CLI*. Umożliwia to dostęp do rejestrów typu licznikowego, rejestrów przechowujących klucze kryptograficzne i parametry dla schematu SHA-1 oraz rejestry techniczne. Zarządzanie i sterowanie pracą układu możliwe jest także zdalnie przez protokół SNMP. W tym celu zdefiniowano bazę MIB oraz wyspecyfikowano drzewo identyfikatorów OID. Odpowiednie oprogramowanie wspiera dostęp lokalny i zdalny.

**Obsługa numerów sekwencyjnych** Ochrona przed atakami *replay* wymaga wprowadzenia numerów sekwencyjnych ramek oraz rozbudowy generatora i weryfikatora HMAC-SHA-1 o obsługę tych numerów. Umożliwia to selektywne akceptowanie tylko tych ramek, które mieszczą się w oknie akceptacji o określonej szerokości. Jako założenie projektowe przyjęto brak współpracy nadajnika z odbiornikiem w zakresie sygnalizacji lub sterowania, która byłaby nieskuteczna na łączach o dużych przepływnościach, lub mogłaby zakłócać swobodny przepływ strumienia danych przez system transmisyjny. Doprowadziło to do znacznej rozbudowy odbiornika (weryfikatora HMAC-SHA-1), który oprócz standardowych sytuacji takich jak przekroczenie zakresu dopuszczalnych wartości numeru sekwencyjnego - "przekręceniełicznika - musi obsługiwać m. in.:

– zagubienie ramki lub bloku ramek,

– odwrócenie naturalnego porządku ramek,
– nadejście ramki ze znacznym opóźnieniem,
– nadejście ramek o wyższych numerach sekwencyjnych niż oczekiwana (wskutek zagubienia jednej lub większej liczby wcześniejszych ramek).

Przypadki te poważnie utrudniają zarządzanie numerami sekwencyjnymi. Wprowadzono okno akceptacji ramek oraz obsługę kilkunastu przypadków szczegółowych.

### 2.4 Środowisko weryfikacyjne

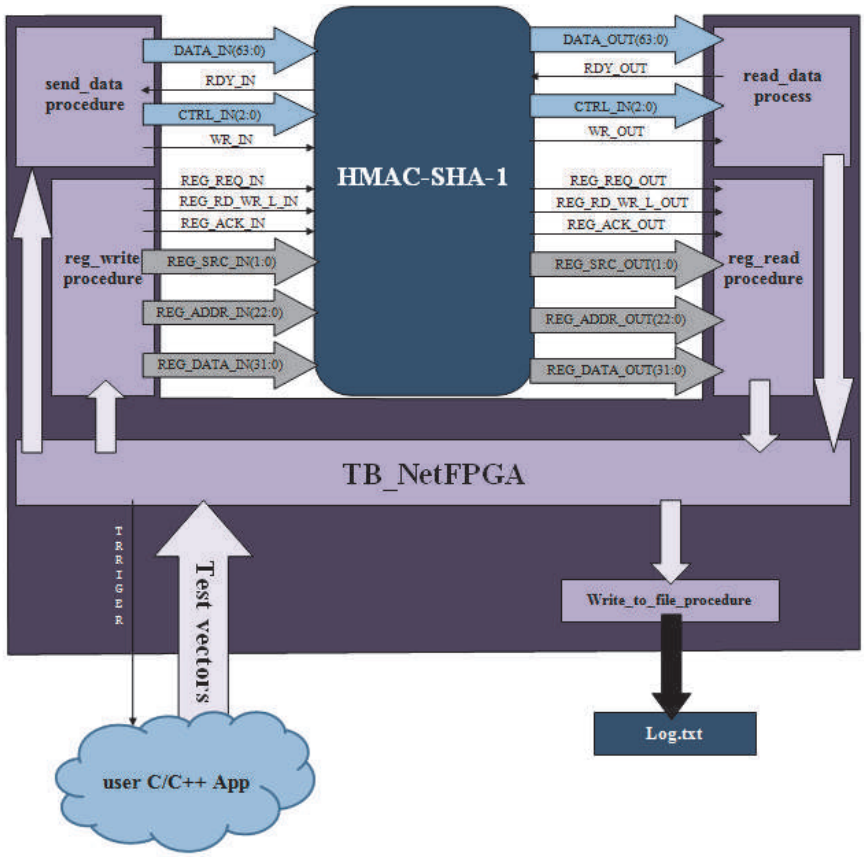
**Diagnostyka przy użyciu procedur testowych** *TestBench* Dla weryfikacji implementacji sprzętowej modułu HMAC-SHA-1 powstało środowisko testowe – rysunek 7. Środowisko to wysyła odpowiednie dane i porównuje odpowiedź modułu HMAC-SHA-1 z danymi referencyjnymi.

Do generowania wektorów testowych stworzono aplikację w języku C/C++ na bazie ogólnodostępnej biblioteki *CryptoPP* [8]. Aplikacja generuje dwa rodzaje ramek Ethernet: przeznaczonych do zabezpieczenia (wysyłane do modułu) oraz zabezpieczonych (referencyjne). Dane następnie są przesyłane za pomocą interfejsu VHPI (*VHDL Procedural Interface*) [9] do środowiska, które wysyła, odbiera i porównuje ramki, a wynik zapisuje do pliku testowego. Powstało kilkadziesiąt scenariuszy testowych które obejmują między innymi:

– wysyłanie ramek o losowej długości i losowej zawartości,
– wysyłanie ramek o określonej długości w pewnym zakresie z krokiem wynoszącym 1 oktet,
– wysyłanie błędnych ramek, na przykład niosących niepoprawny port źródłowy, niepoprawny numer sekwencyjny, złą sygnaturę skrótu (*hash*),
– wysyłanie kolejnej ramki po przetworzeniu przez moduł poprzedniej ramki,
– wysyłanie paczek ramek z konfigurowalnymi odstępami międzyramkowymi.

Testy były przeprowadzane w trybach: *protect* (zabezpieczanie ramek), *verify* (weryfikacja zabezpieczonych ramek) oraz *mixed-mode* (naprzemienne wysyłanie ramek przeznaczonych do zabezpieczenia i ramek zabezpieczonych). Łącznie przetworzono kilkaset tysięcy ramek. W trakcie 20-minutowego testu przetwarzano około 3000 ramek, a w przypadku wykrycia błędów test był powtarzany. Dla przyspieszenia testów do weryfikacji używano dedykowanej karty HES opisanej w dalszej części tego rozdziału.

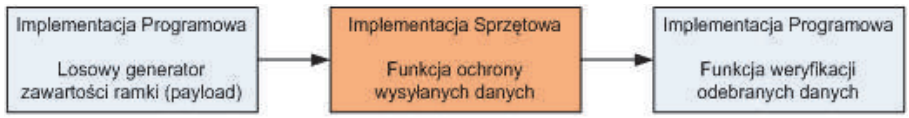
**Weryfikacja programowo-sprzętowa** Przed docelową implementacją na platformie NetFPGA wykonano programową implementację HMAC-SHA-1. Na taki krok zdecydowano się z trzech powodów. Po pierwsze, implementacja programowa przebiega znacznie szybciej niż implementacja wersji sprzętowej. W przypadku błędów implementacyjnych związanych z logiką algorytmu przetwarzania ramki pozwoliło to zaoszczędzić czas potrzebny na implementację poprzez uniknięcie powtórzenia tych błędów w sprzęcie. Po drugie, rozwiązanie to umożliwiło



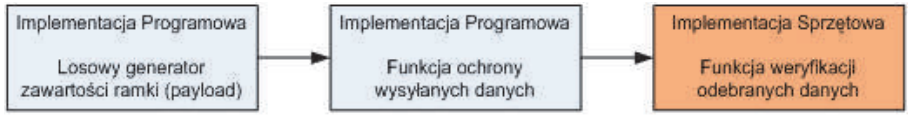

**Rysunek 7.** Schemat blokowy środowiska testowego HMAC (*TestBench*)

weryfikację implementacji sprzętowej przy użyciu implementacji programowej. Wreszcie potwierdzono, że opracowane wcześniej algorytmy są jednoznacznie rozumiane przez dwa zespoły implementatorów, pracujące niezależnie. Wersję programową również przetestowano pod względem poprawności obliczania sygnatury HMAC-SHA-1 zgodnie z dokumentem RFC 2202 [10]. Testy przewidują dwie konfiguracje (por. rysunki 8 i 9), co pozwala zweryfikować poprawność obliczania sygnatury HMAC, formatowania pól ramki oraz zarządzania numerami sekwencyjnymi.

Konfiguracja przedstawiona na rysunku 8 pozwala określić zdolność poprawnej generacji ramki z sygnaturą HMAC i zarządzanie numerami sekwencyjnymi. Druga konfiguracja, rysunek 9, sprawdza sprzętową funkcję weryfikacji ramki. Weryfikacja polega na obliczeniu sygnatury oraz porównaniu jej z otrzymaną w ramce. Następnie testowana jest funkcja kontroli numerów sekwencyjnych oraz




**Rysunek 8.** Schemat blokowy układu testowania weryfikacji zawartości ramki: konfiguracja 1




**Rysunek 9.** Schemat blokowy układu testowania weryfikacji zawartości ramki: konfiguracja 2

przywracanie pierwotnej formy ramki tj. bez pól sygnatury HMAC oraz numerów sekwencyjnych. Środowisko jest monitorowane za pomocą programowego analizatora ruchu sieciowego.

**Uruchomienie i diagnostyka** Moduł HMAC-SHA-1 został zsyntezowany dla układu *Virtex2P 50FF1152* dysponującego 22464 blokami logicznymi (*slice*) i wykorzystuje 95% jego zasobów. Analiza implementacji poparta eksperymentami pozwoliła na sformułowanie wniosku, że wysoka gęstość upakowania logiki może prowadzić do niestabilnej pracy układu przy wyższych przepływnościach bitowych. Wynika to z ograniczeń platformy sprzętowej Virtex II i jest związane z tym, że program z trudem mieści się w układzie Virtex II. Ze względu na rozbudowaną logikę przetwarzania (np. moduł Arbitra, umożliwiający zrównoleglenie przetwarzania), narzędzia nie są w stanie przeprowadzić implementacji projektu w strukturze rozmieszczając bloki logiczne tak, by można było je taktować domyślnym zegarem karty 125MHz. Ograniczenie to wynika ze specyfiki układu Virtex II i przemawia za przeniesieniem pracy do środowiska Virtex V na kartę *netFPGA10G*, gdzie można będzie w całości wykorzystać możliwości wynikające z architektury rozwiązania HMAC dopuszczającej przetwarzanie równoległe. Automatyczna synteza okazała się nieoptymalna. Aby utrzymać standardowe taktowanie architektury NetFPGA (125MHz) została zastosowana ręczna optymalizacja struktury modułu.

Do testowania wydajności układu HMAC-SHA-1 wykorzystano sprzętowo akcelerowany układ testowania implementacji (karta HES) a w dalszym etapie generator Spirent z oprogramowaniem *Test Center*. Modul HMAC-SHA-1 testowany na karcie HES wykazuje poprawne działanie dla wysokiej częstotliwości taktowania. W połączeniu z generatorem Spirent układ HMAC-SHA-1pracuje stabilnie przy przepustowościach zagregowanych do 750 Mbit/s.

**Weryfikacja przy użyciu sprzętowego generatora ruchu** Opisane powyżej metody testowania zostały powtórzone w scenariuszach testowych z użyciem generatora ruchu Spirent. Testy badały wydajność implementacji w różnych warunkach pracy oraz testy odporności na błędy i zakleszczenia. Generator był źródłem strumieni ramek o stałych i zróżnicowanych rozmiarach (naprzemiennie długich i krótkich) oraz niebędących wielokrotnością 8 bajtów, o różnej i zmiennej przepływności bitowej, o różnych odstępach międzyramkowych, o różnej zawartości ramek, a także strumieni generowanych w oparciu o referencyjne scenariusze testowe. Doświadczenia wynikające z tych prac stanowią materiał na osobną publikację.

### 2.5 Wnioski z implementacji

Obliczanie i wprowadzanie do ramki sygnatury skrótu HMAC-SHA-1 oraz numeru sekwencyjnego jest względnie łatwe w implementacji programowej. W przypadku implementacji sprzętowej nakład pracy wymagany do uruchomienia układu oraz na opracowanie procedur testowych, a także nakład pracy wymagany do całościowego przetestowania rozwiązania jest bardzo duży, albowiem lokalizacja i usuwanie błędów są tutaj zdecydowanie trudniejsze. Zastosowanie dobrych praktyk, takich jak układy testbench, interfejs VHPI oraz użycie metody akceleracji sprzętowej HES pomaga w szybszym lokalizowaniu błędów, jednak nie uwalnia od samego poszukiwania błędów. W toku prac projektowych i realizacyjnych przezwyciężono problem niskiej przepustowości układu Virtex i małej dostępności zasobów tego układu. Prace implementacyjne zostały zakończone powodzeniem.

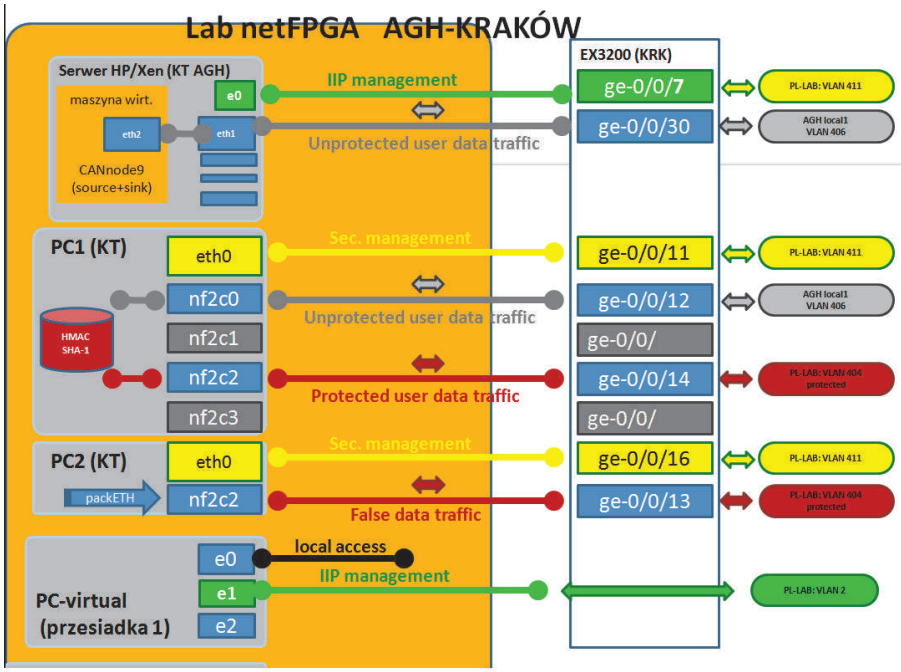
### 2.6 Integracja w środowisku PL-LAB

Układ został włączony do w sieci PL-LAB w celu zabezpieczenia ruchu generowanego przez Równoległy Internet CAN na łączu pomiędzy Krakowem i Wrocławiem. Schemat konfiguracji programowo-sprzętowej węzła PL-LAB w Krakowie dla potrzeb eksperymentów prowadzonych przez zespół *Security* jest pokazany na rysunku 10.

W konfiguracji wykorzystano sieci VLAN obsługiwane przez system PL-LAB i terminowane na odpowiednich portach przełączników dostępowych *EX3200* ulokowanych w Krakowie oraz we Wrocławiu. Strumień danych użytkowych od pochodził od węzła sieci CAN będącego maszyną wirtualną. Dla przenoszenia ruchu zabezpieczonego (*protected user data traffic*) używano dedykowanego VLAN tak, jak to przedstawia rysunek 10.

## 3 Wykrywanie anomalii w szeregach czasowych

Zachowanie węzła Systemu IIP może być analizowane w kontekście charakterystyki zmian wartości wybranych atrybutów opisujących stan węzła. Istotnym elementem umożliwiającym wykrywanie stanów nietypowych (anomalii) węzła




**Rysunek 10.** Schemat blokowy układu testowania weryfikacji zawartości ramki: implementacja programowa

Systemu IIP jest zatem czas obserwacji. Korzystając z tej właściwości, w celu ochrony Systemu IIP została zaadoptowana i zaimplementowana jedna z dostępnych metod wykrywania anomalii w szeregach czasowych. Atrybutami opisującymi działanie węzła Systemu IIP, które mają szczególne znaczenie w kontekście zadania wykrywania zagrożeń są: poziom wykorzystania czasu procesora, poziom zajętości pamięci operacyjnej węzła oraz liczba wysłanych i odebranych ramek dla każdego z Równoległych Internetów. Na przykład nietypowo wysoki poziom obciążenia procesora, czy też nadmiernie duża liczba wysłanych/odebranych ramek w jednostce czasu są charakterystycznymi objawami towarzyszącymi atakom na dostępność usługi (ang. *Denial of Service – DoS*). Takie objawy towarzyszyć mogą również atakom specyficznym dla Systemu IIP, np. polegającym na wstrzykiwaniu ruchu, fałszowaniu ruchu czy też zakłócaniu ruchu poprzez wprowadzanie sztucznych opóźnień. Metodą, która została zaadaptowana na potrzeby zaimplementowanego modułu wykrywania anomalii LAD, jest metoda zaproponowana przez M.Burgessa w pracy [11]. Jedną z istotnych zmian wprowadzonych do oryginalnej koncepcji M.Burgessa było uwzględnienie specyfiki Systemu IIP polegającej na współistnieniu w jednym węźle kilku Równoległych Internetów, które to mogą posiadać odrębną charakterystykę zmienności. Inną istotną modyfikacją istniejącej metody było zintegrowanie procesu wykrywania anomalii

realizowanego przez LAD z działaniami modułu szacującego reputację węzłów Systemu IIP.

Moduł LAD realizuje analizę szeregów czasowych w trzech następujących krokach. Pierwszy krok dotyczy selekcji cechy na potrzeby wykrywania anomalii. Oznacza to, że w tym kroku dokonywany jest wybór najbardziej reprezentatywnego dla potrzeb analizy bezpieczeństwa atrybutu/atrybutów opisującego stan węzła. Krok drugi to szacowanie parametrów. W tym kroku historyczne wartości wybranej cechy porównywane są z wartościami bieżącymi. Ma to na celu oszacowanie poziomu znaczenia zmian wartości obserwowanej cechy dla procesu wykrywania nietypowych stanów węzła. Następnie te dwa kroki są powtarzane wielokrotnie, tak aby ustalić ostateczny model w sposób umożliwiający prawidłowe identyfikowanie stanów nietypowych węzła. Po ostatecznym ustaleniu modelu węzła moduł jest w stanie identyfikować anomalie, które zdefiniowane są jako stany węzła charakteryzujące się znaczącym poziomem rozbieżności pomiędzy bieżącą obserwacją a stanem typowym zdefiniowanym poprzez statystyki wyliczone w fazie ustalania modelu zachowania węzła. Jednym z podstawowych założeń dotyczących konstruowanego modelu jest cykliczność obserwowanych zmian wartości wybranej cechy.

Na potrzeby przedstawienia idei zaimplementowanej metody analizy szeregów czasowych przyjęto, że analizie podlega szereg opisujący liczbę odebranych przez węzeł bajtów danych. Szereg został oznaczony w następujący sposób:

$$X = (x_1, x_2, \ldots, x_l, \ldots), \tag{1}$$

gdzie $x_l$ jest liczbą odebranych bajtów w oknie czasowym o numerze $l$. Przy czym szerokość okna czasowego jest odrębnym parametrem podlegającym optymalizacji ze względu na dokładność wykrywania anomalii. Wartość ta, podobnie jak inne wartości zmienne algorytmu, może być wyznaczona z wykorzystaniem analizy za pomocą krzywych ROC (ang. *Receiver Operating Characteristic*). Metoda ta jest powszechnie wykorzystywana w zadaniach oceny jakości klasyfikacji i w tym przypadku może być również użyta w celu oceny jakości procesu wykrywania anomalii dla przyjętych różnych wartości progowych.

Następnym istotnym elementem metody jest wyróżnienie dwóch podszeregów z analizowanego szeregu $X$. Podszeregi te oznaczone zostały odpowiednio $X_P$ i $X_T$

$$X_{P,l} = \frac{1}{P}\sum_{k=0}^{P-1} x_{l-k}, X_{T,l} = \frac{1}{T}\sum_{k=0}^{T-1} x_{l-kP}, \tag{2}$$

gdzie $P$ oraz $T$ są wartościami wykorzystanymi w celu uśrednienia wartości (wartość $P$ jest nazywana okresem charakterystycznym szeregu $X$). Szeregi $X_P$ i $X_T$ składają się zatem z wartości średnich obserwowanej cechy, wyliczonych na wartościach szeregu $X$ - odpowiednio szereg $X_P$ z wartości średnich elementów pobranych z kolejnych przedziałów równych co do długości okresowi charakterystycznemu $P$, a szereg $X_T$ z wartości średnich elementów $X$ znajdujących się w tej samej fazie okresu charakterystycznego szeregu $X$. Dla obu tak zdefiniowanych podszeregów wyliczane są wartości standardowego odchylenia $\sigma_{P,l}$ oraz $\sigma_{T,l}$

na odpowiednich przedziałach uśredniania dla wartości wyliczonych na podstawie wykładniczej średniej kroczącej $\bar{X}_{P,l}$ i $\bar{X}_{T,l}$ . Ostatecznie lokalne odchylenie obserwowanej wartości cechy wyrażone jest poprzez następującą formułę

$$\delta_{P,l} = |x_l - \bar{X}_{P,l}|, \delta_{T,l} = |x_l - \bar{X}_{T,l}|, \tag{3}$$

Jeżeli obserwacja $x_l$ dotyczyła stanu anomalnego, to następnie obliczany jest potencjalny niekorzystny wpływ tego stanu na poziom bezpieczeństwa monitorowanego węzła. Wpływ ten oznaczany jest przez zmienną *severity*, której aktualna wartość jest przekazywana przez LAD do modułu reputacyjnego oraz która jest szacowana za pomocą wyrażenia:

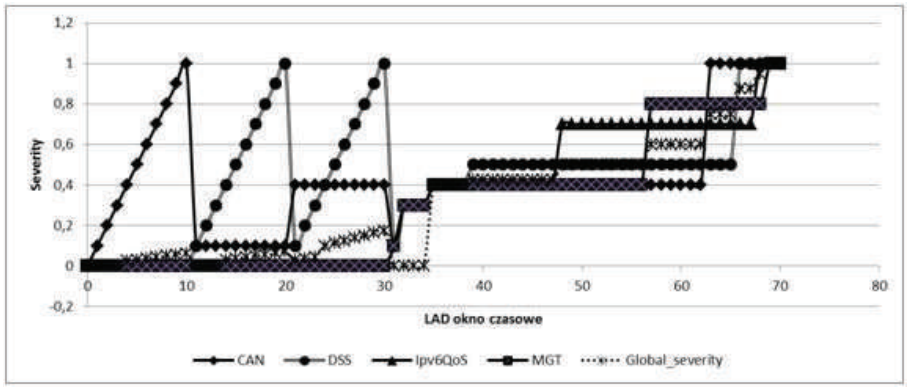
$$c_l = \frac{\sqrt{\left(\delta_{P,l}/\sigma_{P,l}\right)^2 + \left(\delta_{T,l}/\sigma_{T,l}\right)^2}}{3\sqrt{2}}, \tag{4}$$

($c_l = 1$ jeżeli prawa strona równości (4) przekracza 1). Warto zauważyć, iż w przypadku gdy aktualnie zmierzona wartość cechy jest zbliżona do wartości średniej, to jest gdy bieżąca obserwacja koresponduje z doświadczeniem historycznym, to wartość zmiennej *severity* jest bliska 0. Natomiast gdy różnica pomiędzy aktualną wartością cechy oraz jej wartością średnią przewyższa trzykrotnie wartość wyliczonego odchylenia standardowego, to wartość zmiennej *severity* jest maksymalna, równa 1. Drugim, komplementarnym atrybutem określającym poziom zagrożenia wykrytej anomalii jest *intensity*. Wartość tego parametru jest wyliczana jako średnia wartość ze zmiennej *severity* liczona dla charakterystycznego okresu szeregu $X$.

Specyficzną cechą architektury Systemu IIP jest to, że węzły IIP przekazują cztery, w ogólności niezależne, strumienie ruchu sieciowego związane z czterema Równoległymi Internetami (IPv6 QoS, DSS, CAN oraz Management). Dlatego kolejnym istotnym elementem opracowanej metody wykrywania anomalii była możliwość wykrywania anomalii na ogólnym poziomie analizy, tzn. dotyczących całego węzła Systemu IIP, a nie tylko poszczególnych Równoległych Internetów. Oznacza to konieczność zdefiniowania miar *severity* oraz *intensity* z uwzględnieniem wiedzy dotyczącej wszystkich czterech Równoległych Internetów. Globalna wartość *severity* łączy wartości *severity* wyliczone dla wszystkich Równoległych Internetów i jest wyrażona w następujący sposób:

$$\begin{aligned} c_{l-global} = \frac{\gamma}{4} \left(\beta_{DSS} * c_{l-DSS} + \beta_{CAN} * c_{l-CAN}\right. \\ \left. + \beta_{IPv6QoS} * c_{l-IPv6QoS} + \beta_{MGT} * c_{l-MGT}\right), \end{aligned} \tag{5}$$

gdzie wartości *severity* dla poszczególnych Równoległych Internetów są obliczone w sposób przedstawiony wcześniej z wykorzystaniem wiedzy dotyczącej zmian w ruchu w obrębie danego Internetu. Wartości współczynników występujących w równaniu (5) sumują się do jedności i dają możliwość zrównoważenia wpływu poszczególnych Równoległych Internetów na poziom globalnej wartości *severity*. Wartość parametru jest równa liczbie Równoległych Internetów, dla których wartość *severity* przekroczyła predefiniowaną wartość progową. W przykładzie przedstawionym na rysunku 11 przyjęto ten próg na poziomie $0,3$ oraz

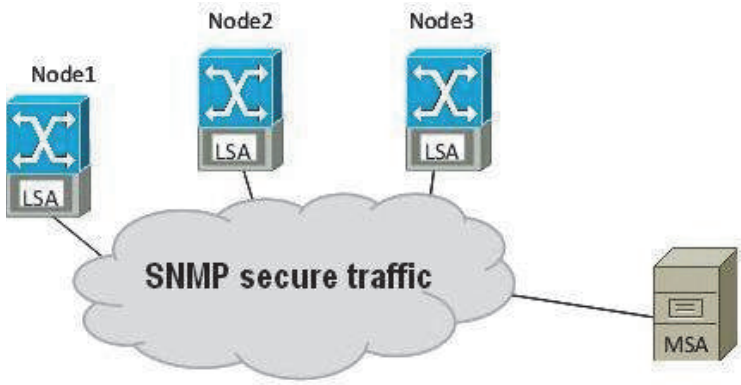



**Rysunek 11.** Poziom globalnej zmiennej *severity* oraz wartości *severity* dla poszczególnych Równoległych Internetów

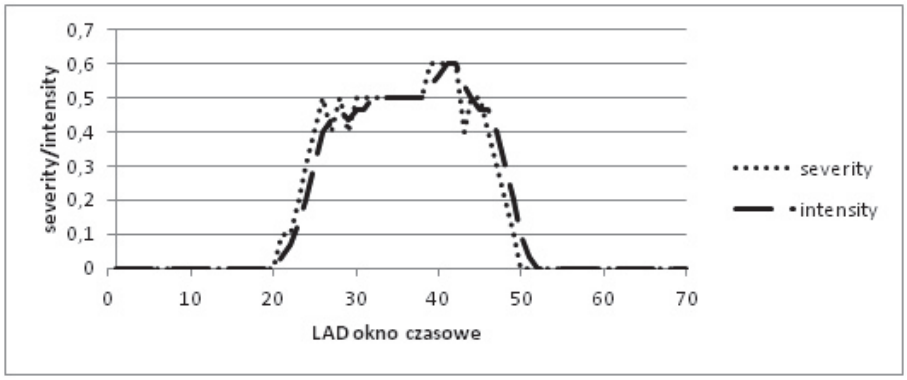
wszystkim parametrom przypisano wartość równą $0,25$. Z postaci wzoru (5) wynika, że nie tylko zmiany aktualnego poziomu *severity* dla poszczególnych Równoległych Internetów mają wpływ na ostateczną wartość globalną. Zależy ona również od liczby Równoległych Internetów, których praca została zaburzona w znaczącym stopniu - im więcej, tym mocniej ich zaburzenia oddziałują na globalną ocenę bezpieczeństwa węzła.

Poniżej został przedstawiony scenariusz testowy, który demonstruje możliwości zaimplementowanego w module LAD algorytmu analizy szeregów czasowych na potrzeby detekcji specyficznych ataków poziomu 2 architektury Systemu IIP (IIPS Level 2). Tymi specyficznymi atakami są: wstrzykiwanie ruchu (ang. *traffic injection*) oraz zaburzanie ruchu (ang. *ruffling*). Moduł LAD analizuje szeregi czasowe utworzone z wartości opisujących liczbę bajtów odebranych w ramach danego Równoległego Internetu w badanym węźle Systemu IIP. Wartości te są dostarczane do LAD co 10 sekund przez podsystem monitorowania Systemu IIP za pomocą metody Get dostępnej w ramach protokołu SNMP dla odpowiednio ustalonej wartości OID. Na potrzeby badań ruch był generowany w sposób sztuczny z wykorzystaniem aplikacji D-ITG [12].

Pierwszy eksperyment dotyczył symulowanych ataków, które mają miejsce w obrębie pojedynczego Równoległego Internetu i dotyczyły sytuacji wstrzykiwania ruchu, gdzie atak był realizowany z poziomu węzła Node3 przeciw węzłowi Node2 (rys. 12). W początkowym okresie Node3 generował ruch typowy o charakterystyce odstępów pomiędzy kolejnymi IIPS-PDU zgodnie z rozkładem Pareto, gdzie wartość średnia dla rozkładu została ustalona na poziomie 15 ms a wartość odchylenia standardowego na poziomie 75 ms. Natomiast rozmiar przesyłanej ramki został zdefiniowany jako równy 5000 B. Moduł LAD w węźle Node2 analizuje szereg czasowy utworzony z wartości oznaczających liczbę odebranych bajtów w kolejnych oknach czasowych. Temu okresowi eksperymentu odpowiada wartość 0 dla zmiennej *severity* przedstawionej na wykresie na ry-



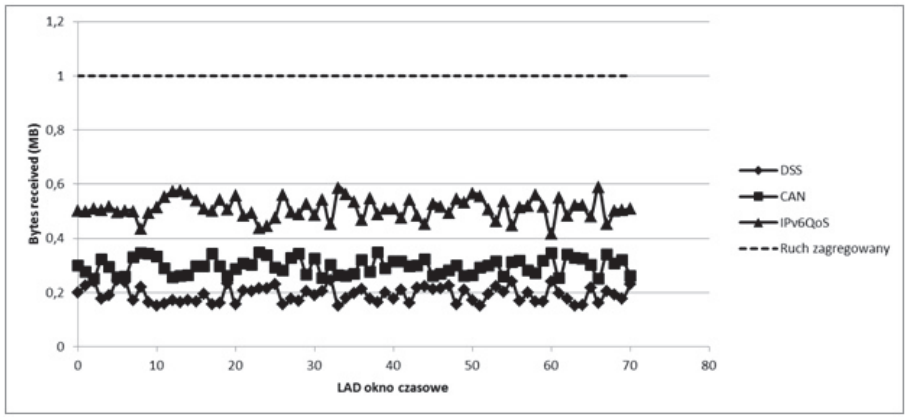

**Rysunek 12.** Środowisko dla testów integracji systemu bezpieczeństwa i systemu zarządzania uruchomione w Instytucie Łączności w Warszawie



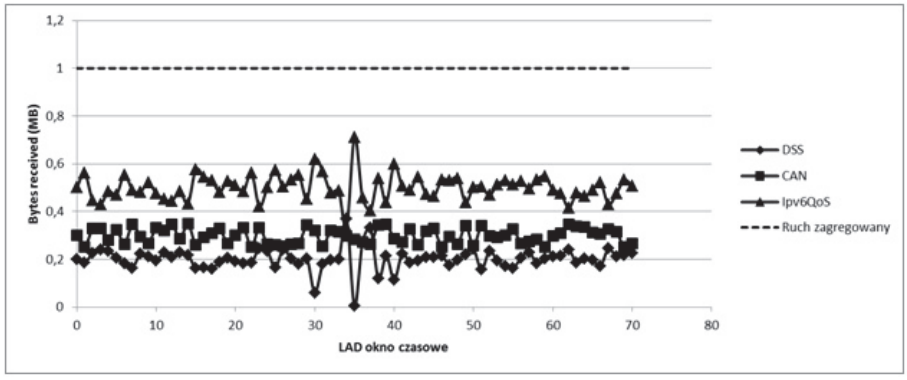

**Rysunek 13.** Wykrywanie ataków typu *traffic injection*

sunku 13. Wstrzyknięcie ruchu przez węzeł Node3 zostało zasymulowane poprzez wysłanie strumienia dodatkowych ramek z interwałami losowanymi z rozkładu normalnego z wartością średnią równą 20 ms oraz odchyleniem standardowym 5 ms. Dodatkowe ramki miały rozmiar 500 B. Wstrzykiwanie dodatkowego ruchu rozpoczęto po upływie 20 okien. LAD w węźle Node2 aktualizuje swoje obserwacje i w miarę narastania liczby zaburzonych elementów obserwowanego szeregu czasowego zwiększa się również wartość zmiennej *severity*. Na rysunku 13 można również zauważyć, iż po pewnym czasie LAD uczy się nowej charakterystyki obserwowanego ruchu i wartość *severity*/*intensity* ponownie spada do 0.

Następny scenariusz testowy pokazuje różnice pomiędzy wykrywaniem anomalii w zagregowanym ruchu pochodzącym z wszystkich Równoległych Inter-




**Rysunek 14.** Ruch bez anomalii




**Rysunek 15.** Ruch z anomaliami w Równoległych Internetach DSS oraz IPv6QoS

netów, a wykrywaniem anomalii w ruchu każdego z Równoległych Internetów z osobna. Na rysunkach 14 oraz 15 średnia liczba bajtów zagregowanego ruchu kierowanego w danym oknie czasowym do węzła IIPS jest stała. Oznacza to, że w obu przypadkach gdy poddajemy analizie ruch zagregowany, nie zostaną wykryte żadne anomalie, chociaż charakterystyka ruchu na poziomie poszczególnych Równoległych Internetów w sytuacjach zilustrowanych na rysunkach 14 i 15 jest różna. Stan ten ulegnie zmianie, gdy będzie rozpatrywany oddzielnie ruch dla każdego z Równoległych Internetów. W tym przypadku zostanie wykryte anomalne natężenie ruchu dla Równoległego Internetu DSS oraz IPv6QoS pomiędzy 30. a 40. oknem czasowym (rysysunek 15). Dzięki temu podana wcześniej miara globalnej wartości *severity* dla węzła będzie w stanie również odzwierciedlić różnicę obu przedstawionych przypadków.

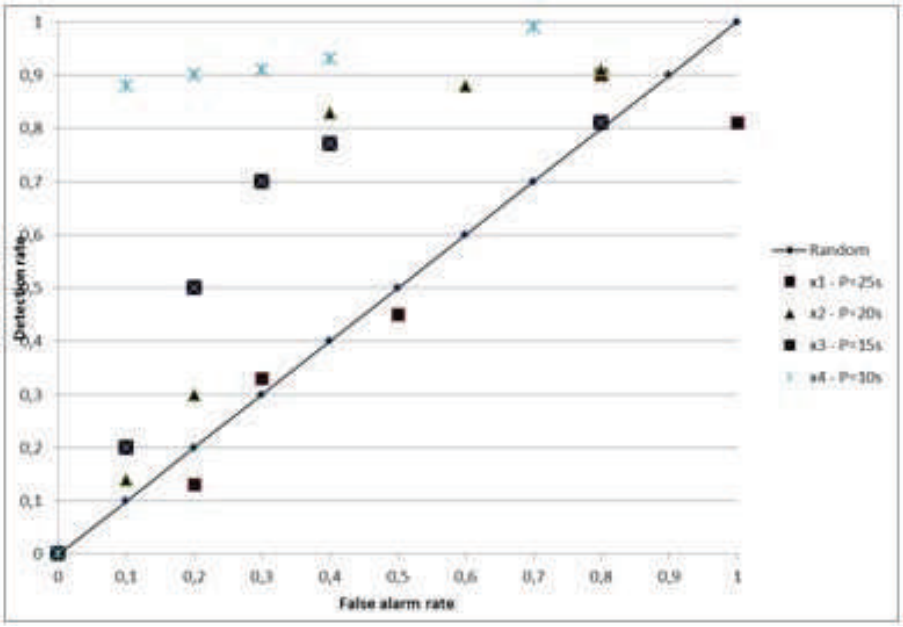
Zanim opisana metoda będzie mogła być wykorzystana w rzeczywistym Systemie IIP, konieczna jest odpowiednia parametryzacja algorytmu. Jedną z częściej wykorzystywanych metod pozwalających na ustalenie optymalnych wartości parametrów algorytmu klasyfikującego zdarzenia jako normalne lub anomalne jest metoda wykorzystująca krzywe ROC (ang. *Receiver Operating Characteristic*) [13]. Krzywe te umożliwiają obrazowe porównanie współczynnika fałszywych alarmów z współczynnikiem wykrywalności zdarzeń nietypowych. Współczynnik fałszywych alarmów jest ilorazem liczby błędnych alarmów i liczby wszystkich anomalii, które miały miejsce w badanym okresie, a współczynnik wykrywalności jest ilorazem liczby prawidłowo wykrytych anomalii i wszystkich anomalii w badanym okresie. Na przykład, aby dobrać odpowiednią wartość okresu charakterystycznego $P$ dla równania (2) zostało przeprowadzonych pięć eksperymentów z różnymi wartościami $P$. W eksperymentach wykorzystano przedstawiony wcześniej scenariusz ataku poprzez wstrzyknięcie ruchu. Otrzymane krzywe ROC zostały przedstawione na rysunku 16. Uzyskane wyniki wskazują na to, iż odpowiednią dla tego typu zdarzeń wartością $P$ jest okres o długości 10 sekund, gdyż dla takiej wartości okresu charakterystycznego przy niewielkim współczynniku fałszywych alarmów otrzymano najwyższy współczynnik wykrywalności sięgający 90% . Analogiczne eksperymenty powinny być przeprowadzone również dla określenia właściwych wartości pozostałych parametrów zmiennych wykorzystywanych w zaimplementowanej metodzie analizy szeregów czasowych.

Ponieważ System IIP na obecnym etapie jest w trakcie ustawicznego rozwoju oraz badań laboratoryjnych, w przedstawionym przykładzie posłużono się ruchem generowanym w sposób sztuczny, za pomocą dodatkowych aplikacji. Docelowo parametryzacja algorytmu powinna wykorzystywać dane pochodzące z rzeczywistego Systemu IIP.

## 4 Wykrywanie anomalii metodami opartymi na eksploracji danych

Wykrywanie anomalii za pomocą metod eksploracji danych bazuje na analizie rejestru SRE. Wpisy SRE mogą przykładowo zawierać: nieudane logowanie, odrzucenie PDU przez moduł HMAC, próba odczytania poprzez SNMP klucza z bazy MIB, do którego użytkownik nie ma stosownych uprawnień czy też odrzucenie połączenia w sieci zarządzania niezgodnego z polityką bezpieczeństwa. Z każdym z wymienionych zdarzeń, oprócz jego typu i czasu wystąpienia, związane są pewne atrybuty, np. nazwa konta, adresy IP odrzuconego PDU: źródłowy i docelowy, czy OID klucza w bazie MIB. SRE generowane przez różne programy działające pod kontrolą systemu Linux w węźle IIP są logowane za pomocą standardowego podsystemu syslog, skąd moduł LAD pobiera dane do analizy. Takie rozwiązanie umożliwi w przyszłości łatwe uwzględnianie innych zdefiniowanych SRE. Aktualnie moduł LAD przystosowany jest do analizowanie danych z następujących źródeł:

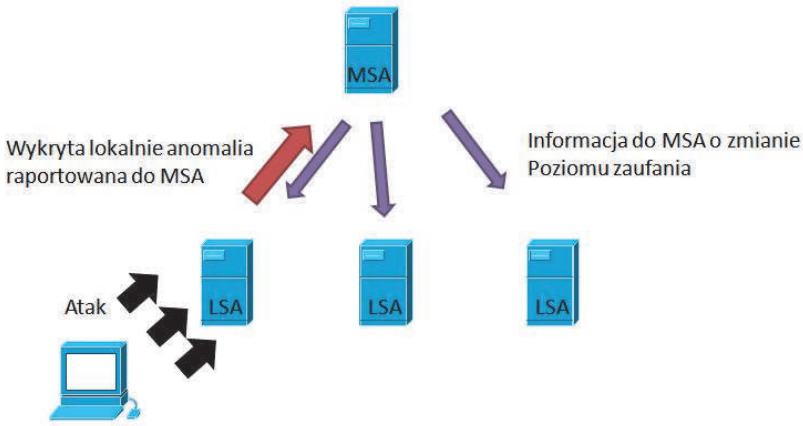
– zapora ogniowa systemu Linux (*ip6tables*),




**Rysunek 16.** Krzywe ROC dla różnych wartości $P$

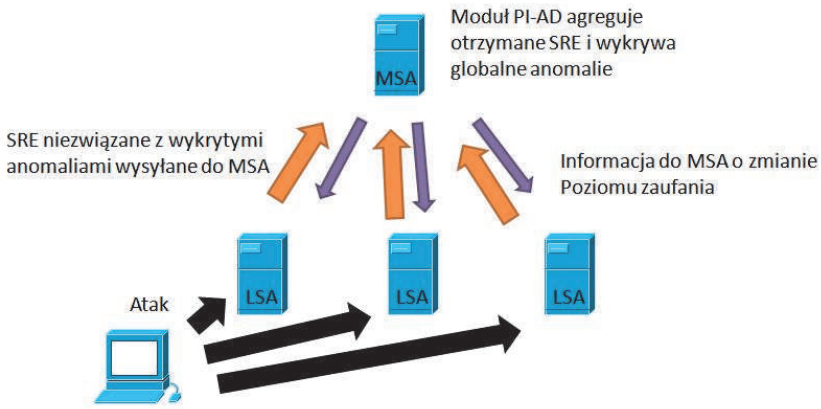
– proxy SNMP opisane szczegółowo w podrozdziale 6, pozwalające uzyskać logi związane z atakami na serwer SNMP,
– interfejs diagnostyczny modułu HMAC,
– logi z serwera SSH.

Proponowana metoda opiera się na wykrywaniu tzw. zbiorów częstych (*frequent sets*), tj. powtarzających się często wzorców zachowań [14]. Jeśli dane do analizy traktujemy jako zbiory odpowiednich atrybutów, to zbiorem częstym nazywamy podzbiór występujący co najmniej *minSup*-krotnie w analizowanych danych. Parametr *minSup*, nazywany minimalnym wsparciem, jest parametrem wejściowym algorytmu wyszukiwania zbiorów częstych. Zbiory częste można wykrywać za pomocą różnych algorytmów. W ramach projektu został zaimplementowany algorytm *a priori*, którego dokładny opis można znaleźć w [5]. Analizując tą metodą SRE dotyczące np. odrzuconych prób dostępu do kluczy bazy MIB można wykryć wzorzec związany z próbą poznania przez danego użytkownika wartości pewnych gałęzi. Jeśli w wykrytym zbiorze częstym będzie znajdowała się informacja o adresie IP, oznaczać to będzie iż ataki były przeprowadzane z jednej maszyny. Brak w wykrytym zbiorze częstym identyfikatora OID świadczy o próbie dotyczy poznania różnych gałęzi. W sytuacji kiedy wykrytym zbiorze częstym będzie atrybut związany z OID, a brak będzie atrybutu związanego z adresem IP, atak był przeprowadzany z wielu maszyn i dotyczył określonego elementu bazy MIB. W wyniku analizy uzyskanych zbiorów częstych, do modułu




**Rysunek 17.** Działanie modułu lokalnego wykrywania anomalii (LAD)

obliczania reputacji (*reputation calculator*) zostanie przekazana ocena ryzyka (*severity*) związanego z daną anomalią oraz prawdopodobieństwo trafności oceny dokonanej przez moduł LAD. Informacje te wykorzystywane są przez system zarządzania reputacją (szczegółowo omówiony w podrozdziale 5). W celu wykrywania szerokiego spektrum ataków, analiza anomalii z wykorzystaniem metod eksploracji danych wykorzystywana jest na dwóch poziomach - lokalnym poprzez moduł LAD (ang. *local anomaly detection*) oraz globalnie poprzez moduł PI-AD (ang. *Parallel Internet-wide Anomaly Detection*). Moduł LAD zainstalowany na każdym węźle Systemu IIP analizuje wszystkie zarejestrowane SRE i wykrywa w nich zbiory częste. Wykrycie zbioru częstego jest dowodem wystąpienia pewnego powtarzalnego wzorca, który z dużym prawdopodobieństwem może być atakiem, lub inną anomalią, która wymaga interwencji administratora. Jeśli wykryty zbiór częsty posiada element opisujący adres nadawcy, generowany jest komunikat o anomalii i wysyłany za pomocą modułu LSA do modułu centralnego MSA. Na podstawie poszczególnych elementów zawartych w wykrytym węźle, jak i otrzymanego wsparcia wykrytego zbioru częstego obliczane są wymagane przez podsystem obliczania reputacji parametry wykrytej anomalii: jej potencjalne zagrożenie (*severity*) oraz prawdopodobieństwo trafności (*intensity*). Sytuacja tego typu przedstawiona jest na rysunku 17. Działania podejmowane w trakcie ataku powodują wygenerowanie wystarczającej liczby zdarzeń SRE aby wykryć zbiór częsty. Analiza wykrytego zbioru częstego prowadzi do wygenerowania anomalii, która później raportowana jest do modułu centralnego MSA. Na podstawie tych informacji, moduł MSA wylicza poziom zaufania danego węzła, a informacja zwrotnie zostaje wysłana do wszystkich węzłów Systemu IIP. Jednak tego typu podejście nie chroni przed atakującymi stosującymi bardziej finezyjne metody ataków, przykładowo atakowanie wielu maszyn w ten sposób, że żadna pojedyn-




**Rysunek 18.** Działanie modułu globalnego wykrywania anomalii (PI-AD)

czo nie wykrywa ataku. Tego typu zagrożenia wykrywane na poziomie globalnym poprzez moduł PI-AD. Moduł LAD po wykryciu zbiorów częstych dokonuje przeglądu wszystkich zdarzeń SRE, zarejestrowanych w danym oknie czasowym. Wszystkie zdarzenia SRE, które nie wspierają żadnego z wykrytych zbiorów częstych (to znaczy nie są nadzbiorami jakiegokolwiek wykrytego zbioru częstego) zostają wysłane do dalszej analizy przez moduł PI-AD. Moduł ten, umieszczony w węźle centralnym, agreguje otrzymane SRE ze wszystkich węzłów Systemu IIP. Na zagregowanych danych dokonywane jest wykrywanie zbiorów częstych. Ich analiza pozwala wykryć anomalie analogicznie do operacji przeprowadzanych w module LAD. Rysunek 18 przedstawia działanie modułu PI-AD. Atakujący wysyła pewien ruch do wielu węzłów Systemu IIP. Ponieważ ruch jest bardzo mały, żaden z węzłów lokalnie nie wykrywa zagrożenia, natomiast rejestruje pojedyncze zdarzenia SRE. Wszystkie tego typu zdarzenia wysyłane są za pomocą LSA do modułu PI-AD. Tutaj po dokonaniu agregacji wykrywane są zbiory częste. Pojedyncze z punktu widzenia jednego węzła zdarzenia po zagregowaniu ujawniają powtarzalny wzorzec i pozwalają wykryć anomalie. Pozostałe kroki są analogiczne, jak w poprzednim przykładzie. Komunikat o wykrytej anomalii zostaje wysłany do podsystemu obliczania reputacji, który obniża poziom zaufania do atakującego. Uzyskana w ten sposób informacja zostaje rozesłana do wszystkich węzłów Systemu IIP.

## 5 System reputacyjny

Ujednolicone podejście do problematyki bezpieczeństwa w Systemie IIP stało sie możliwe dzięki wykorzystaniu systemów zaufania i reputacji jako najwyższej warstwy architektury bezpieczeństwa. Reputacja pełni tu funkcję jednolitej miary poziomu bezpieczeństwa węzłów i łączy, uwzględniającej zarówno zagro-

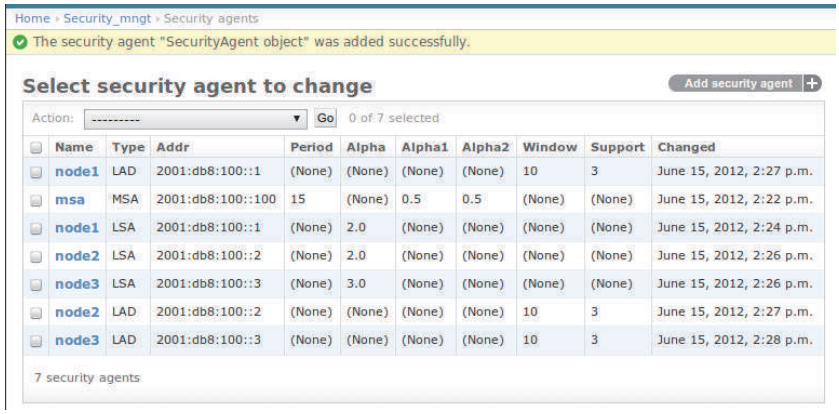
żenia "twarde", przeciw którym stosowane są zwykle kryptograficzne metody ochrony (w naszym przypadku: HMAC, zabezpieczenia protokołu SNMP oraz SSH), jak i "miękkie", wykrywane metodami detekcji anomalii. Kluczową decyzją poprzedzającą propozycję nowej architektury bezpieczeństwa dla Systemu IIP było rozstrzygnięcie, czy wartość reputacji węzła ma być ustalana na zasadzie rekomendacji, czy też obliczana na podstawie obserwacji ruchu pochodzącego z danego węzła przez węzły sąsiednie. W pierwszym przypadku (model rekomendacyjny) ustalanie poziomu zaufania i reputacji danego węzła następuje na podstawie subiektywnych ocen dokonywanych okresowo przez węzły sąsiednie (w naszym przypadku: LSA rezydujące w tych węzłach). Oceny te wykorzystują doświadczenie węzłów komunikujących się z węzłem ocenianym nabyte w ustalonym czasie lub w całej historii pracy sieci. W modelu tym bardzo często jest uwzględniana informacja uzyskana pośrednio, pochodząca od innych węzłów albo z niezależnych źródeł (np. ogólne informacje o zagrożeniach w sieci), por. [15]. W drugim przypadku (model walidacyjny) poziom zaufania i reputacji są obliczane jedynie na podstawie obserwacji zagrożeń Systemu IIP dokonywanych przez LSA w poszczególnych węzłach. Z punktu widzenia całego Systemu IIP wyniki te mają więc charakter subiektywny, jednak oparte są na bezpośrednich danych pomiarowych, czyli danych pochodzących z pierwszej ręki. W zaproponowanej architekturze bezpieczeństwa przyjęto walidacyjny model obliczania zaufania i reputacji i zaadaptowano go w scentralizowanej strukturze zaufania i reputacji obejmującej wszystkie węzły Systemu IIP. W naszym modelu poszczególne LSA raportują swoje, obliczone na podstawie obserwacji ruchu, oceny węzłów sąsiednich do centralnego MSA, który z kolei konsoliduje pozyskane w ten sposób dane i oblicza globalny poziom reputacji wszystkich węzłów. Spośród wielu znanych metod identyfikacji i konsolidacji danych reputacyjnych (por. np. [16]) przyjęto probabilistyczny model bayesowski, w którym na ocenę danego węzła składają się ważone opinie jego sąsiadów (ich wagi sa proporcjonajne do ich aktualnego poziomu reputacji), a płynność zmiany poziomu reputacji jest zagwarantowana przez uwzględnienie czynnika historycznego. Tak zaprojektowany system reputacyjny stanowiący element podsystemu bezpieczeństwa dla IIPS ma dwie podstawowe zalety. Po pierwsze, jest odporny na liczne ataki, na które podatne są systemy oparte na rekomendacji, por. [15]. Po drugie, pozostawia on LSA rozlokowanym w węzłach pewną autonomię przy obliczaniu wartości zaufania i reputacji ich sąsiadów, pozostawiając w kompetencji eksperta zarządzającego centralnie siecią LSA jedynie ustalanie parametrów wykorzystywanych w procedurze obliczania poziomu zaufania (zakres uwzględniania historycznych wartości reputacji, długość okna próbkowania, wartości *severity* i *intensity* dla poszczególnych zagrożeń, itp., por. [4], [5]. W efekcie pozwala to na sprawne i płynne obliczanie poziomu zaufania i reputacji w lokalnym otoczeniu każdego z węzłów oraz niemal natychmiastowa reakcje na dynamicznie zmieniający się poziom zagrożeń w sieci.

System reputacyjny zaprojektowany dla ochrony Systemu IIP jest przeznaczony do wykrywania zagrożeń i ataków w warstwie 2 (Level 2). Korzysta z oryginalnych modułów programowych i sprzętowych oraz ze standardowych metod

ochrony zaimplementowanych w podsystemie bezpieczeństwa Systemu IIP, to znaczy: modułów LAD (wykrywających ataki pochodzące z węzłów Systemu IIP i skierowane przeciwko innym węzłom Systemu IIP), sprzętowego modułu zabezpieczającego HMAC (umożliwiającego wykrywanie ataków pojawiających się na łączu między dwoma sąsiednimi węzłami Systemu IIP), a także z zabezpieczeń kryptograficznych protokołu SNMP i tuneli realizowanych z wykorzystaniem protokołu SSH (chroniących komunikację zarządzania Systemu IIP, podsystemu bezpieczeństwa Systemu IIP i komunikację innych protokołów, np. sygnalizację Równoległego Internetu CAN). Reasumując, zaprojektowany system reputacyjny pozwala oceniać poziom zaufania zarówno samych węzłów, jak i połączeń międzywęzłowych warstwy 2 (Level 2), a także dodatkowo poziom zaufania węzłów i połączeń systemu zarządzania.

Przyjęta scentralizowana architektura systemu reputacyjnego sprawia, że każdy LSA obserwuje ruch przychodzący od węzłów z nim sąsiadujących i przewiduje wystąpienie zagrożeń estymując wartość prawdopodobieństwa (*severities*) wystąpienia wykrytych anomalii, tak jak to już zostało wcześniej przedstawione tym rozdziale (por. także [5]). Wyniki uzyskane w takiej procedurze są okresowo przesyłane do MSA, a tam podlegają procedurze konsolidacji w celu obliczenia wartości poziomu zaufania i reputacji wszystkich węzłów Systemu IIP. Scentralizowana struktura systemu reputacyjnego umożliwia efektywne obliczanie tych wielkości oraz nadzorowanie zachowania wszystkich węzłów Systemu IIP w skali globalnej, co pozwala wykrywać ataki niezauważalne z lokalnej perspektywy jednego lub kilku sąsiadujących węzłów. Wadą takiej scentralizowanej architektury jest fakt, iż MSA staje się naturalnym punktem podatnym na atak odmowy usługi (np. DoS lub DDoS), awarii lub fizycznego odcięcia połączenia. Zagrożeniu temu można jednak przeciwdziałać proponując scenariusze awaryjne korzystające z faktu, iż wszystkie węzły lokalnie dysponują zgromadzona wiedzą na temat poziomu zaufania i reputacji swoich sąsiadów i w razie awarii MSA nadal prowadzą obserwację i bieżącą ocenę swoich sąsiadów. Można zatem zastosować procedurę odtworzenia MSA, używają np. podejścia zaproponowanego w pracy [17]. Wymaga ona w pierwszej kolejności wyboru węzła, w którym zostanie umieszczony nowy MSA, a następnie przesłania lokalnych informacji reputacyjnych zgromadzonych w każdym z LSA umieszczonych w węzłach Systemu IIP do MSA. W krótkim czasie nowy MSA jest w stanie zrekonstruować swoje globalne rekomendacje reputacyjne dla węzłów Systemu IIP poprzez konsolidację danych uzyskanych z LSA.

Jak to zostało przyjęte w architekturze bezpieczeństwa Systemu IIP, każdy LSA estymuje poziom zaufania do wszystkich swoich sąsiadów wykorzystując dwa podejścia: weryfikację ramek Systemu IIP (IIPS-PDU) poprzez sprawdzanie poprawności HMAC oraz wykonywanie algorytmów prowadzących do wykrywania anomalii w lokalnym module LAD. Konsolidując dane uzyskane w weryfikacji wykorzystującej HMAC oraz z poszczególnych algorytmów wykrywania anomalii można uzyskać liczbowe wartości poziomu zaufania połączeń międzywęzłowych. W szczególności, na podstawie takiej analizy można ustalić, czy źródłem wykrytych anomalii ruchu przychodzącego z sąsiedniego węzła jest niewła-




**Rysunek 19.** Panel do konfiguracji MSA, LSA i LAD w architekturze bezpieczeństwa Systemu IIP

ściwe funkcjonowanie węzła (spowodowane na przykład atakiem na ten węzeł), czy też zagrożenie występujące na trasie przesyłanych pakietów (występujące w pewnym miejscu ścieżki między węzłami). Inną przesłanką wystąpienia ataku mogłoby być zaobserwowanie nietypowego ruchu w sieci (na przykład, zalew nieoczekiwanym strumieniem danych) przy równoczesnym braku wskazania przez lokalny LAD niewłaściwego działania któregokolwiek sąsiedniego węzła. Przeciwnie, jeśli źródłem zaobserwowanych zaburzeń jest jeden z węzłów Systemu IIP, to analiza skonsolidowanych rekomendacji przeprowadzona przez MSA prowadzi do potwierdzenia hipotezy o niepoprawnym działaniu węzła. Taka sytuacja ma zazwyczaj miejsce, gdy negatywne rekomendacje (niskie wartości poziomu zaufania) nadchodzą od wszystkich sąsiadów komunikujących się z ocenianym węzłem.

System reputacyjny wyposażony został w interfejs zarządzania w postaci panelu dostępnego poprzez aplikację WWW (por. rysunek 19). Panel ten daje administratorowi możliwość ustawienia wszystkich parametrów zarówno indywidualnego LSA jak i modułu centralnego (MSA), wpływając tym samym na ustawienia globalne systemu reputacji. Dla poszczególnych typów modułów (LSA-/MSA) parametry te to długość okresu próbkowania, parametry formuły obliczenia reputacji oraz adres sieciowy węzła, por. [5]. Dodatkowo panel zarządzający ma możliwość konfiguracji parametrów podsystemu LAD, wpływając na częstość wykrywania oraz ocenę zdarzeń przez ten system. W obecnej implementacji wszystkie powyższe parametry mogą być ustalane za pomocą dołączonego do systemu interfejsu graficznego. Ustalanie wartości parametrów następuje eksperymentalnie, zgodnie z wiedzą ekspercką oraz długofalową obserwacją funkcjonowania systemu, w szczególności jego reakcji na dane klasy naruszeń bezpieczeństwa. Dzięki tym wartościom można uczynić system reputacyjnym mniej lub bardziej czułym na zidentyfikowane klasy ataków, natomiast poprzez

ustalenie czasu próbkowania możliwe jest sterowanie czasem reakcji systemu. Prezentowany interfejs WWW dla architektury bezpieczeństwa Systemu IIP, został zaprojektowany jako przyjazny użytkownikowi końcowemu oraz intuicyjny w obsłudze, przy jednoczesnym minimalnym wpływie działania interfejsu na strukturę węzłów w sieci. Aplikacja może być umiejscowiona praktycznie w dowolnym punkcie sieci. Została stworzona przy użyciu pakietu *Django* [18], który jest wysokopoziomowym, otwartym i darmowym frameworkiem przeznaczonym do tworzenia aplikacji internetowych. Zarówno *Django*, jak i *pySNMP* [19] (pakiet użyty do implementacji bezpiecznego systemu wieloagentowego) napisane są w języku Python, stąd ich kompatybilność jest wysoka, a integracja oraz eksploatacja ułatwione. Główne okno opisywanej aplikacji, prezentuje dostępne w systemie moduły LSA, MSA oraz LAD wraz z wyszczególnionymi ich nazwami, adresami sieciowymi, parametrami oraz czasem ostatniej ich zmiany. Z uwagi na dynamiczny charakter sieci, w każdym momencie administrator systemu może dodać lub usunąć węzeł z panelu zarządzania. Aplikacja w prosty oraz naturalny sposób umożliwia zmianę parametrów węzłów architektury bezpieczeństwa Systemu IIP.

Gdy wartość wybranego parametru zostanie zaktualizowana przez administratora, aktualna konfiguracja musi zostać przesłana do danego węzła. W tym celu *host*, na którym zainstalowana jest aplikacja WWW, wysyła wiadomość aktualizująca do węzła systemu bezpieczeństwa, którego konfiguracja uległa zmianie. Węzeł po otrzymaniu tej wiadomości aktualizuje swoją konfigurację. Powyższa komunikacja odbywa się z pomocą bezpiecznego kanału, zrealizowanego przez zaszyfrowane oraz uwierzytelnione wiadomości *Inform* protokołu SNMPv3. Następnie aktualna konfiguracja zapisana zostaje w lokalnej (z punktu widzenia aplikacji WWW) bazie danych. Obecna implementacja korzysta z relacyjnej bazy danych, zarządzanej za pomocą systemu *SQLite* [20]. Dostęp do opisywanego systemu zarządzania WWW zabezpieczony jest w podstawowy sposób, tj. za pomocą uwierzytelnienia użytkowników na podstawie ich nazw (login) oraz haseł, jednak w przyszłej, rozszerzonej wersji planowane jest zabezpieczenie tego dostępu za pomocą protokołu *HTTPS* (certyfikowany SSL).

## 6 Zabezpieczenie systemu zarządzania siecią i zarządzania systemem bezpieczeństwa

Jak wspomniano wcześniej (por. też [5]), elementy systemu reputacyjnego komunikują się wykorzystując protokół SNMPv3. Do zabezpieczenia procesu komunikacji systemu bezpieczeństwa (głównie systemu reputacyjnego), jak również pozostałej komunikacji ruchu zarządzania, został przedstawiony moduł *SNMP-PROXY*. Jest to nowy element architektury bezpieczeństwa Systemu IIP, który został zintegrowany z częścią systemu zarządzania IIP. Funkcjonalność realizowana przez *SNMP-PROXY* zaimplementowana została dla osiągnięcia dwóch celów. Pierwszym z nich jest dostarczenie usług bezpieczeństwa dla ruchu zarządzania Systemu IIP. Drugim celem jest udostępnienie podsystemu logowania

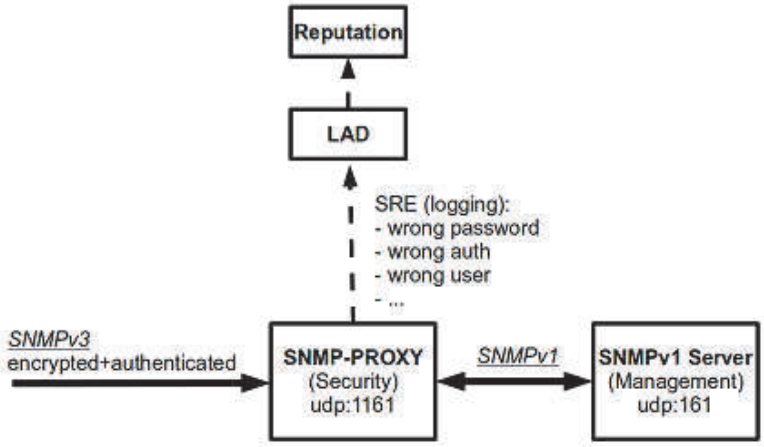
(logowanie SRE), który związany jest z ruchem SNMP. Architektura *SNMP-PROXY* zaprezentowana została na rysunku 20.

Zabezpieczenie ruchu zarządzania Systemu IIP osiągnięte jest za pomocą użycia protokołu SNMPv3 w miejsce protokołów o niższych wersjach (które są w stanie zagwarantować znacznie niższy poziom bezpieczeństwa). *SNMP-PROXY* oczekuje na połączenia, wymuszając na komunikujących się podmiotach użycie bezpiecznego protokołu wraz z uwierzytelnieniem za pomocą nazw użytkowników oraz haseł. Cały ruch SNMP zostaje wówczas zaszyfrowany oraz uwierzytelniony. Po udanym nawiązaniu połączenia otrzymane zapytanie SNMP przekazywane jest do właściwego serwera zarządzania SNMP, gdzie jest przetwarzanie oraz obsługiwane. Komunikacja między *SNMP-PROXY* a właściwym serwerem SNMP jest komunikacją lokalną i może odbywać się za pomocą niezabezpieczonych wersji protokołu. Jednak kluczowym aspektem dla funkcjonowania opisywanej architektury jest wymuszenie tego, by cały ruch kierowany do serwera SNMP przechodził przez opisywany moduł. *PROXY-SNMP*, z uwagi na wykorzystanie protokołu SNMPv3, może wymagać maksymalnie trzech parametrów bezpieczeństwa: nazwy bezpieczeństwa (najczęściej utożsamianej z nazwą użytkownika) i dwóch haseł używanych odpowiednio do szyfrowania oraz uwierzytelniania komunikatów. W naszej obecnej implementacji połączenie jest ustanowione wtedy i tylko wtedy, gdy wszystkie te parametry maja właściwie ustawione wartości. Moduł *SNMP-PROXY* został zaimplementowany z wykorzystaniem biblioteki *pySNMP* [19], która zezwala na użycie wybranych metod kryptograficznych. Jako algorytm kodów uwierzytelniających wiadomości dostępny jest schemat HMAC bazujący na funkcjach skrótu MD5 lub SHA-1. Natomiast do realizacji funkcji szyfrowania może zostać użyty jeden z następujących szyfrów: DES, 3DES oraz AES (z kluczami o wybranej długości bitowej: 128, 192, 256).

W celu dostarczenia informacji o zdarzeniach bezpieczeństwa dla ruchu SNMP (logowanie SRE), *SNMP-PROXY*, przetwarzając zapytania SNMP, loguje wszystkie podejrzane aktywności, które następnie są przekazywane do modułu wykrywania anomalii. Zdarzeniami, które są logowane i mogą być następnie rozpatrywane jako ataki są: błędna nazwa użytkownika, błędne hasło szyfrowania lub uwierzytelnienia, wersja protokołu inna niż SNMPv3, próba transmisji nieszyfrowanej lub nieuwierzytelnionej oraz inny zestaw modułów kryptograficznych niż ustalony przez administratora. Zdarzenia bezpieczeństwa logowane są w ustalonym formacie za pomocą funkcji systemowej *syslog()*. Pojedynczy wpis zawiera informacje o czasie zdarzenia, adresie źródłowym, porcie źródłowym, typie błędu wraz z dostępnym kontekstem protokołu SNMP (takim jak np. żądane zasoby, które nieuprawiony użytkownik chciał uzyskać).

## 7 Integracja podsystemu bezpieczeństwa w laboratorium PL-LAB

Opisane w poprzedzającej części tego rozdziału elementy podsystemu bezpieczeństwa zostaną zintegrowane z innymi elementami Systemu IIP w środowisku

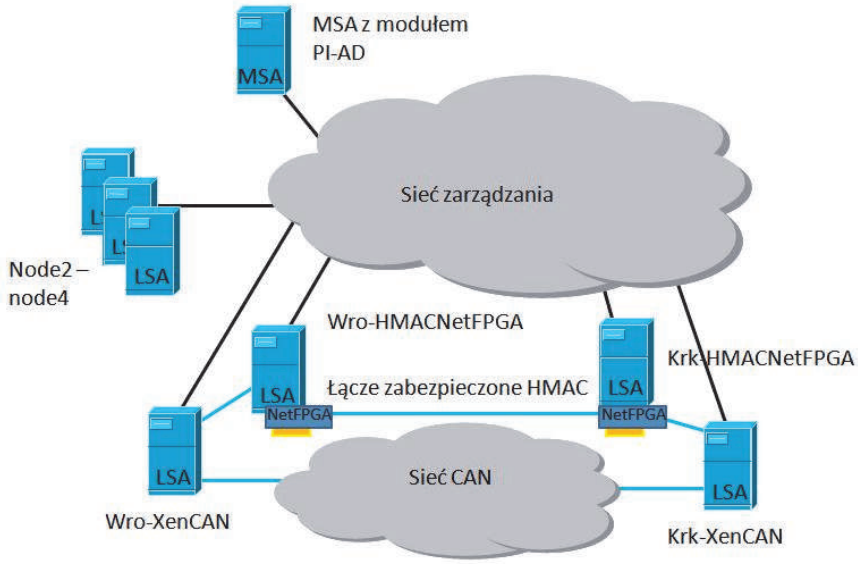



**Rysunek 20.** Architektura zabezpieczenia SNMP-PROXY

zdalnego laboratorium PL-LAB. Rysunek 21 przedstawia proponowaną topologię połączeń oraz wykorzystywane elementy. Zaproponowana topologia pozwoli na integrację wszystkich elementów podsystemu bezpieczeństwa, jak również integrację z innymi elementami Systemu IIP. Pozwoli to na przeprowadzenie nowych eksperymentów, podczas których zostaną dobrane parametry opisywanych metod tak, aby wykrywać istotne z punktu widzenia Systemu IIP anomalie.

Najważniejszym elementem zaproponowanej integracji jest połączenie dwóch węzłów Systemu IIP łączem zabezpieczonym za pomocą modułu HMAC. Węzłami terminującymi zabezpieczony ruch będą węzły Systemu IIP zrealizowane w technologii Xen uruchomione we Wrocławiu oraz Krakowie. Moduł HMAC, zaimplementowany na kartach NetFPGA zostanie uruchomiony w obu lokalizacjach na dedykowanych maszynach. Dodatkowo w lokalizacji Wrocław zostanie uruchomiony w środowisku wirtualnym centralny węzeł zarządzania z oprogramowaniem MSA i PI-AD oraz dodatkowo trzy węzły z modułami LSA i LAD. Podjęto już prace zmierzające do uruchomienia na węzłach Xen maszyn internetu CAN oraz podłączenia ich do już uruchomionej w laboratorium PL-LAB sieci CAN. Pozwoli to wygenerować na łączach tych maszyn ruch zgodny ze specyfiką Systemu IIP.

Na aktualnym etapie prac wszystkie siedem (poza MSA) węzłów posiada zainstalowane oprogramowanie systemu reputacyjnego (agenta LSA) oraz moduły wykrywania anomalii LAD, w oparciu o obie metody. W celu umożliwienia pobierania danych przez modułu LAD na tych węzłach zostały zainstalowane i odpowiednio skonfigurowane demony rsyslog oraz snmpd. Na ósmym węźle został uruchomiony moduł obliczania reputacji MSA, jak też moduł globalnego wykrywania anomalii - PI-AD. Dodatkowo prowadzone są prace zmierzające do uruchomienia oprogramowania HMAC na kartach NetFPGA oraz integracji in-




**Rysunek 21.** Topologia sieci eksperymentalnej integrującej wszystkie podsystemy bezpieczeństwa w sieci PL-LAB

terfejsu diagnostycznego HMAC z modułem LAD. Wszystkie przedstawione na rysunku 21 węzły są podłączone do podsieci zarządzania laboratorium PL-LAB.

## 8 Wnioski

Przedstawiono aktualny zarys prac implementacyjnych nad architekturą bezpieczeństwa zorientowaną na przeciwdziałanie zagrożeniom na poziomie 2 Systemu IIP. W toku prac wykonano i wstępnie przetestowano programową oraz sprzętową realizację modułu generującego oraz weryfikującego sygnaturę skrótu HMAC-SHA-1, dołączaną do ramki System IIP w celu ochrony przed atakami przeciwko systemowi transmisyjnemu Systemu IIP, a także implementacje modułów wykrywania anomalii i systemu reputacyjnego. Wykorzystując niewielkie środowisko testowe z systemem komunikacyjnym zorganizowanym w oparciu o protokół SNMPv3 dokonano szeregu eksperymentów mających na celu weryfikację skuteczności mechanizmów bezpieczeństwa w obecności wybranych metod ataków. Zwrócono uwagę na niektóre aspekty implementacji mogące mieć wpływ na dalsze prace integracyjne w obrębie Systemu IIP. W obecnej chwili trwają końcowe prace nad integracją mechanizmów bezpieczeństwa z Systemem IIP poprzez ogólnopolską sieć testową PL-LAB.

## Literatura


1. Cabaj K. i in., Implementation and testing of Level 2 security architecture for the IIP System, Przegląd Telekomunikacyjny - Wiadomości Telekomunikacyjne, Vol.85(81), No.8-9, pp.1426-1435, 2012.
2. Konorski J. i in., Implementacja sprzętowa modułu HMAC-SHA-1 do ochrony komunikacji w Systemie IIP, Przegląd Telekomunikacyjny - Wiadomości Telekomunikacyjne, Vol.85(81), No.8-9, pp.1418-1425, 2012.
3. Burakowski W., Tarasiuk H., and Beben A., System IIP for supporting „Parallel Internets (Networks)", FIA meeting, Ghent 2010, fi-ghent. fiweek.eu/files/2010/12/1535-4-System-IIPFIA-Ghent-ver1.pdf
4. Cabaj K. i in., Architektura bezpieczeństwa Systemu IIP na poziomie wirtualizacji zasobów, Przegląd Telekomunikacyjny - Wiadomości Telekomunikacyjne, Vol.84(80), No.8-9, pp.846-851, 2011.
5. Cabaj K. i in., Architektura bezpieczeństwa Systemu IIP, rozdział w: W. Burakowski, P. Krawiec [Eds.], Inżynieria Internetu Przyszłości, Vol.1, pp.43-60, Warszawa 2012.
6. Konorski J. i in., A Virtualization-Level Future Internet Defense-in-Depth Architecture, in: Communications in Computer and Information Science (CCIS), vol.335, Recent Trends in Computer Networks and Distributed Systems Security, Part 1, pp. 283-292, Springer-Verlag, Berlin Heidelberg 2012.
7. Netlink Protocol Library Suite (libnl). http://www.infradead.org/ tgr/libnl/.
8. Crypto++® Library 5.6.1. http://www.cryptopp.com/.
9. IEEE. 1076-2008 IEEE Standard VHDL Language Reference Manual. 2009. ISBN:978-0-7381-5800-6.
10. RFC 2202. Test Cases for HMAC-MD5 and HMAC-SHA-1", September 1997.
11. Burgess M., Two dimensional time-series for anomaly detection and regulation in adaptive systems, Proc. IFIP/IEEE 13th Int. Workshop on Distributed Systems: Operations and Management, DSOM 2002, pp. 169-185
12. D-ITG Distributed Internet Traffic Generator, http://www.grid.unina. it/software/ITG/
13. Green D.M., Swets J.A., Signal detection theory and psychophysics, John Wiley and Sons, New York 1966.
14. Agrawal R., Srikant R., Fast algorithm for mining association rules, In: J.B. Bocca, M. Jarke, and C.Zaniolo, [ed]. Proceedings of 20th International Conference on Very Large Databases, pp. 487-499, 1994.
15. Konorski J., Orlikowski R., DST-Based Detection of Noncooperative Forwarding Behavior of MANET and WSN Nodes, Proc. 2nd Joint IFIP WMNC., Gdansk, Poland, 2009.
16. Ciszkowski T. et al., Towards Quality of Experience-based Reputation Models for Future Web Service Provisioning, Telecommunication Systems, Vol.51, No.4, pp.283-295, 2012.
17. Oryńczak G., Kotulski Z., Notary-based self-healing mechanism for centralized peer-to-peer infrastructures, Annales UMCS Informatica AI XII, 1, pp.97–112, 2012.
18. https://www.djangoproject.com/
19. http://pysnmp.sourceforge.net/
20. http://www.sqlite.org/

# Specyfikacja i implementacja modułu wymiarowania zasobów w Systemie IIP

Krzysztof Wajda[1], Mirosław Kantor[1], Janusz Gozdecki[1], Jacek Rak[2]

[1] AGH Akademia Górniczo-Hutnicza w Krakowie
[2] Katedra Teleinformatyki, Politechnika Gdańska

**Streszczenie** Artykuł zawiera opis specyfikacji oraz implementacji modułu wymiarowania zasobów działającego w warstwach L1 i L2 architektury Systemu IIP. Omówiono zadania modułu wymiarowania, zdefiniowano styki z systemem zarządzania oraz Równoległymi Internetami, jak również zaproponowano algorytmy optymalizacyjne oraz przedstawiono przykładowe wyniki działania tych algorytmów.

## 1 Wprowadzenie

W ramach realizacji Systemu IIP [1]-[4] w projekcie Inżynieria Internetu Przyszłości (IIP), składającego się z czterech warstw: fizycznej (L1), wirtualizacji (L2), Równoległych Internetów (RI) - L3 oraz sieci wirtualnych (L4), jest wspierane działanie trzech Równoległych Internetów: IPv6_QoS (*Quality of Service*), CAN (*Content Aware Networks*) i DSS (*Data Streams Switching*). Obecnie, na etapie implementacji systemu testuje się różne rozwiązania wirtualizacyjne umożliwiające rozdział zasobów fizycznych na środowiska wirtualizacyjne. Istotnym elementem funkcjonalnym umożliwiającym taki rozdział jest moduł wymiarowania (ang. *provisioning*), współpracujący z systemem zarządzania. Niniejszy artykuł, odnosząc się do specyfikacji modułu wymiarowania i reguł współpracy z systemem zarządzania przedstawionych w [5], opisuje przygotowane modele optymalizacyjne oraz dokumentuje przykładowe wyniki określające przydział zasobów Systemu IIP L1/L2 dla poszczególnych RI. Moduł wymiarowania spełnia rolę mediatora w Systemie IIP: na podstawie wymagań na zasoby przekazane przez administratorów poszczególnych RI oraz dostępnych informacji na temat konfiguracji węzłów i łączy następuje walidacja realizowalności wymagań, rozdział zasobów elementarnych pomiędzy RI oraz zostaje przeprowadzona konfiguracja urządzeń. Stan sieci i konfiguracja urządzeń są dostępne dzięki uaktualnianiu informacji w bazie danych systemu zarządzania. Problem przydziału zasobów fizycznych do tworzonych sieci wirtualnych jest rozważany w literaturze publikowanej w ostatnich latach. Przykładowa metodologia wymiarowania sieci wirtualnych została zaproponowana w [6], gdzie autorzy opisują zarówno aspekty teoretyczne, jak i praktyczne dotyczące wydajności wyznaczania rozwiązania. Ponieważ analizowane zagadnienie jest problemem NP-trudnym, zatem

w celu efektywnego wyznaczenia rozwiązania, zostały zaproponowane algorytmy heurystyczne. Problem statycznego przydziału zasobów do sieci wirtualnych, tj. przydziału bez rekonfiguracji, był analizowany m.in. w [7]. Autorzy skorzystali w tym przypadku z algorytmu najkrótszej ścieżki w celu oceny ścieżek dostępnych na poziomie fizycznym, mogących potencjalnie tworzyć strukturę sieci wirtualnych. Heurystyki umożliwiające odwzorowanie kolejno węzłów i łączy wirtualnych zaproponowano w [8]-[9], natomiast algorytm określający równocześnie przydział węzłów i łączy wirtualnych opisano w [10]. Z kolei w [11] został zaproponowany algorytm umożliwiający przydział zasobów sieci fizycznej dla węzłów i łączy wirtualnych żądających określonych parametrów. Celem tej heurystyki, która uwzględnia również wymagania węzłów ukrytych, tj. węzłów znajdujących się na ścieżce fizycznej, na której jest realizowane łącze wirtualne, jest maksymalizacja wolnej przepustowości oraz dostępnych zasobów obliczeniowych sieci fizycznej.

Zawartość niniejszego artykułu jest następująca: w rozdziale 2 zaprezentowano trzy modele optymalizacyjne definiujące sposób podziału zasobów elementarnych pomiędzy RI według różnych kryteriów. Rozdział 3 przedstawia specyfikację sieci testowej oraz przykładowe wyniki uzyskane dla tej sieci. Rozdział 4 zawiera podsumowanie oraz naświetla dalsze plany z zakresu realizacji modułu wymiarowania.

## 2 Prezentacja modeli wymiarowania zasobów

Poniżej zostaną przedstawione trzy modele optymalizacyjne zaimplementowane w module wymiarowania warstw L1/L2 Systemu IIP. Zagadnienie wymiarowania jest nadal rozwijane, ponieważ prace nad implementacją Systemu IIP skutkują weryfikacją ustaleń dotyczących sposobu definiowania zasobów dla poszczególnych RI oraz zbioru parametrów przekazywanych do platform sprzętowych. Istotnym aspektem modelu wymiarowania jest również przyjęta funkcja celu. Definiując zaproponowane poniżej modele, przyjęto że:

- węzły sieci obejmują zarówno węzły dostępowe, jak i szkieletowe,
- dla każdego RI macierz ruchu (definiująca strumienie ruchu dla par węzeł źródłowy-węzeł docelowy) jest znana *a priori*,
- związek pomiędzy przenoszonym ruchem a niezbędnymi zasobami węzłowymi jest liniowy.

Model 1 zawiera podstawowe sformułowanie problemu wymiarowania z funkcją celu dotyczącą minimalizacji wykorzystania zasobów łączy. Model 2 ujmuje w sposób bezpośredni wykorzystanie zasobów węzłowych, takich jak np. moc obliczeniowa. Model 3 zawiera dodatkowe ograniczenia na opóźnienie transmisji na ścieżkach, istotne dla niektórych klas usług. Topologia sieci jest reprezentowana przez graf skierowany $G = (V, E)$, gdzie $V$ oznacza zbiór węzłów, natomiast $E$ reprezentuje zbiór łączy (łuków skierowanych). Przepustowość każdego łącza jest wyrażona w Mbit/s i jest wielkością ciągłą.

**Model 1:**
**Celem jest minimalizacja użycia zasobów transmisyjnych (przepustowości łączy) przy spełnieniu podstawowych wymagań dotyczących doboru tras (*routing*).**

**Indeksy:**

| | |
|---|---|
| $i = 1, 2, 3$ | instancje poszczególnych RI (DSS, IPv6_QoS i CAN) |
| $t = 1, 2, \ldots, \lvert T\rvert$ | węzły tranzytowe (szkieletowe) |
| $e = 1, 2, \ldots, \lvert E\rvert$ | łącza w sieci (łuki skierowane) |
| $d = 1, 2, \ldots, \lvert D\rvert$ | żądania dla $i$-tego RI |
| $v = 1, 2, \ldots, \lvert V\rvert$ | węzły sieci |
| $V \setminus T$ | zbiór węzłów brzegowych (dostępowych) |

**Stałe:**

$a_{ev}$ przyjmuje wartość 1, jeżeli węzeł $v$ jest początkowym dla łuku $e$; wartość 0 w innym przypadku
$b_{ev}$ przyjmuje wartość 1, jeżeli węzeł $v$ jest końcowym dla łuku $e$; wartość 0 w innym przypadku
$c_e$ całkowita przepustowość łącza $e$
$\gamma_{ie}$ minimalna część przepustowości żądanej dla $i$-tego RI na łączu $e$
$h_{id}$ wartość żądania $d$ dla $i$-tego RI (w Mbit/s)
$s_{id}$ węzeł źródłowy dla żądania $d$ $i$-tego RI
$u_{id}$ węzeł docelowy dla żądania $d$ $i$-tego RI

**Zmienne (ciągłe):**

$x_{iev} \geqslant 0$ przepustowość zarezerwowana dla $i$-tego RI na łączu $e$ wychodzącym z węzła $v$ (w Mbit/s)
$z_{iev} \geqslant 0$ przepustowość przydzielona na łączu $e$ dla żądania $d$ $i$-tego RI (w Mbit/s)

**Funkcja celu:**

Zminimalizować całkowitą przepustowość wykorzystaną w celu transmisji żądanego ruchu:

$$\textbf{minimize} \quad F = \sum_i \sum_e \sum_v x_{iev} \tag{1}$$

**Ograniczenia**

$$\sum_v a_{ev}x_{iev} = \sum_v b_{ev}x_{iev} \qquad i \in I \quad e \in E \tag{2}$$

$$\sum_e b_{et}x_{iet} = \sum_e a_{et}x_{iet} \qquad i \in I \quad t \in T \tag{3}$$

$$\sum_v a_{ev}x_{iev} \geqslant \gamma_{ie}c_e \qquad i \in I \quad e \in E \tag{4}$$

$$\sum_i \sum_v a_{ev}x_{iev} \leqslant c_e \qquad e \in E \tag{5}$$

$$\sum_e a_{ev}z_{ied} - \sum_e b_{ev}z_{ied} = \begin{cases} h_{id} & \text{dla } v = s_{id} \\ -h_{id} & \text{dla } v = u_{id} \\ 0 & \text{inne} \end{cases} \quad i \in I \; V \in V \; d \in D \tag{6}$$

$$\sum_v \sum_d a_{ev}z_{ied} \leqslant x_{iev} \qquad i \in I \quad e \in E \tag{7}$$

Warunek (2) ma zapewnić, że strumień ruchu wychodzący z węzła $v$ przez łącze $e$ dla $i$-tego RI jest wprowadzany do węzła na drugim końcu łącza $e$. Równanie (3) realizuje postulat ciągłości ruchu (tzn. warunek wyrażający "prawo Kirchhoffa") dla węzłów tranzytowych. Warunek (4) zapewnia, że wielkość przepustowości zaalokowanej na łączu $e$ dla $i$-tego RI przewyższa minimalny próg żądany dla $i$-tego RI na łączu $e$. Formuła (5) zapewnia, że całkowita przepustowość przydzielona na łączu $e$ dla $i$-tego RI nie przekracza całkowitej przepustowości tego łącza. Warunek (6) zapewnia odpowiednią obsługę każdego przepływu pomiędzy węzłami źródłowym i docelowym każdego żądania $d$. Spełnienie warunku (7) z kolei gwarantuje, że całkowity przepływ dla wszystkich żądań dla $i$-tego RI na łączu $e$ jest równy całkowitej przepustowości zaalokowanej dla $i$-tego RI na wskazanym łączu.

Jako alternatywę dla funkcji celu zdefiniowanej wzorem (1), zastosowano również jej zmodyfikowaną postać daną równaniem (8) w celu maksymalizacji całkowitej dostępnej przepustowości:

$$\textbf{maximize} \quad F = \sum_e (c_e - \sum_v \sum_i x_{iev}) \tag{8}$$

---

**Model 2:**
**Celem jest minimalizacja wykorzystania zasobów transmisyjnych (przepustowości łączy) rozszerzona o minimalizację użycia zasobów węzłowych przy spełnieniu podstawowych wymagań doboru tras (*routing*).**

---

Model 2 jest rozwinięciem Modelu 1 z dodatkowymi warunkami dotyczącymi optymalizacji rozdziału zasobów dostępnych w węzłach. W modelu są wprowadzone dodatkowe wymagania na zasoby węzłowe (oprócz zasobów łączy), co skutkuje dodatkowymi ograniczeniami dla procesu optymalizacji. W poniższym sformułowaniu są więc uwzględnione dodatkowe warunki dotyczące przydziału mocy obliczeniowej dostępnej w węzłach. Dodatkowo inne zasoby, takie jak pamięć masowa, RAM czy wielkość buforów mogą być traktowane w analogiczny sposób.

**Indeksy:**
Lista indeksów jest taka sama jak dla Modelu 1.

**Stałe:**
Lista stałych jest taka sama jak dla Modelu 1 i dodatkowo obejmuje następujące elementy:

$\theta_{iev}$ moc obliczeniowa na jednostkę przepustowości dla $i$-tego RI oraz łącza $e$ wychodzącego z węzła $v$

$\delta_{iev}$ moc obliczeniowa na jednostkę przepustowości dla $i$-tego RI oraz łącza $e$ wchodzącego do węzła $v$

$\xi_v$ całkowita moc obliczeniowa dostępna w węźle $v$

**Zmienne (ciągłe):**

$y_{iv} \geqslant 0$ zasoby zarezerwowane dla celów przetwarzania strumieni ruchu dla $i$-tego RI oraz łącza $e$ w węźle $v$ (np. w Mflops)

**Funkcja celu:**

Zminimalizować całkowitą moc obliczeniową niezbędną w celu transmisji żądanego ruchu:

$$\textbf{minimize} \quad F = \sum_v \sum_i \sum_i y_{iv} \tag{9}$$

**Ograniczenia**
Ograniczenia (2)-(7) pozostają w mocy plus dodatkowo:

$$y_{iv} = \sum_e \theta_{iev} a_{ev} x_{iev} + \sum_e \epsilon_{iev} b_{ev} x_{iev} \qquad i \in I \quad v \in V \tag{10}$$

$$\sum_i y_{iv} \leqslant \xi_v \qquad v \in V \tag{11}$$

Wyznaczenie całkowitej mocy obliczeniowej zastosowanej w węźle $v$ dla celów przetwarzania strumieni ruchu dla $i$-tego RI oraz łącza $e$ jest realizowane przez równanie (10). Z kolei, warunek (11) gwarantuje, że całkowita moc obliczeniowa zastosowana w węźle $v$ nie może być większa niż całkowita moc obliczeniowa dostępna w tym węźle.

---

**Model 3:**
**Rozwinięcie Modelu 2 z wprowadzonymi dodatkowymi ograniczeniami dotyczącymi opóźnień na ścieżkach transmisyjnych.**

---

Model 3 wprowadza dodatkowe ograniczenia na maksymalne opóźnienie transmisji dla każdego strumienia. Jeżeli dla konkretnego strumienia nie ma ograniczenia na opóźnienie, można przypisać w tym przypadku odpowiednio dużą

wartość ograniczenia, aby uczynić opóźnienie czynnikiem nieistotnym w procesie obliczeniowym.

**Indeksy:**
Lista indeksów jest taka sama jak dla Modelu 1.

**Stałe:**
Lista stałych jest taka sama jak dla Modelu 1 i 2 oraz dodatkowo zawiera:

- $f_e$ maksymalne opóźnienie transmisji wzdłuż łącza $e$
- $n_{ied}$ przyjmuje wartość 1, jeżeli żądanie $d$ $i$-tego RI realizowane jest przez łącze $e$; wartość 0 w przeciwnym przypadku
- $g_{id}$ maksymalne dopuszczalne opóźnienie transmisji dla żądania $d$ $i$-tego RI pomiędzy węzłami końcowymi transmisji

**Zmienne (ciągłe):**
Lista zmiennych jest taka sama jak dla Modelu 1.

**Funkcja celu:**
Zminimalizować całkowitą przepustowość użytą w celu transmisji żądanego ruchu (podobnie jak w Modelu 1):

$$\textbf{minimize} \quad F = \sum_v \sum_e \sum_i x_{iev} \tag{12}$$

przy spełnieniu warunków (2)-(7), (10)-(11) oraz dodatkowo ograniczenia (13) gwarantującego, że całkowite opóźnienie transmisji dla żądania $d$ dla $i$-tego RI jest mniejsze, niż założona wartość graniczna $g_{id}$:

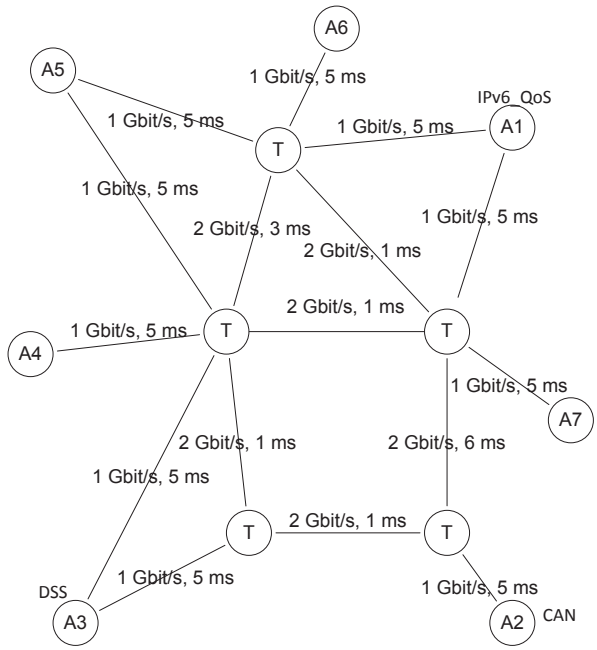
$$\sum_e n_{ied} \leqslant g_{id} \qquad i \in I \quad d \in D \tag{13}$$

Stosując inne definicje funkcji celu wraz ze specyfikacją warunków modelu można otrzymać różne sformułowania problemu optymalizacji przydziału zasobów. Przedstawione powyżej trzy modele są przykładami rozwiązań wdrażanych w ramach projektu IIP.

## 3 Weryfikacja charakterystyk modułu wymiarowania zasobów

Proces wymiarowania definiuje topologię dla każdego RI. Na podstawie informacji o dostępnych zasobach, lokalizacji węzłów oraz wymagań co do wydajności poszczególnych RI, proces ten określa, które węzły i łącza wirtualne będą stanowiły topologię wymiarowanego RI oraz jaką część zasobów węzła należy przypisać dla danego RI. W celu porównania wyników działania zaproponowanych algorytmów wymiarowania zostały przeprowadzone obliczenia dla przykładowej sieci przedstawionej na rysunku 1 zawierającej dwa typy węzłów: węzły brzegowe (*edge* lub *access*, oznaczone jako $A$) oraz węzły szkieletowe (*transit*, oznaczone

jako $T$). Rysunek 1 zawiera również oznaczenia dotyczące typu węzłów brzegowych, dedykowanych dla poszczególnych RI, tzn. IPv6_QoS, DSS oraz CAN. Te oznaczone węzły (IPv6_QoS, DSS oraz CAN) wspierają tylko jeden typ ruchu wprowadzonego/odebranego, podczas gdy inne węzły brzegowe wspierają wszystkie typy RI. Ruch tranzytowy jest wspierany przez wszystkie węzły szkieletowe. Przykładowo, dla ruchu RI typu CAN strumienie są przesyłane pomiędzy węzłami wskazanymi jako CAN oraz pomiędzy wszystkimi nieetykietowanymi węzłami brzegowymi.




**Rysunek 1.** Przykładowa topologia i szczegółowe parametry 12-węzłowej sieci zastosowanej w obliczeniach

Rysunek 1 wskazuje dodatkowo przepustowość oraz opóźnienie transmisyjne każdego łącza. Przepustowość łącza jest wielkością ciągłą (tzn. niemodularną). Dokładne parametry żądań (strumieni ruchu) użyte w obliczeniach są zamieszczone w Tabeli 1. W celu sprawdzenia elastyczności Modelu 3 założono, że RI typu IPv6_QoS nie nakłada żadnych dodatkowych ograniczeń na maksymalne opóźnienie transmisji. W tym przypadku przyjęto dużą wartość opóźnienia (250 ms), aby uczynić opóźnienie czynnikiem nieistotnym w procesie obliczeniowym.

Wyniki działania dla proponowanych trzech modeli zostały przedstawione w Tabeli 2. Zobrazowano cztery typy parametrów wyjściowych: poziom wyko-

**Tablica 1.** Szczegóły dotyczące żądań (strumieni ruchu) użyte w obliczeniach

| Strumień ruchu (z → do) | Typ RI | Przepustowość [Mbit/s] | Maks. dopuszczalne opóźnienie [ms] |
|---|---|---|---|
| 1 → 4 | IPv6_QoS | 200 | 250 |
| 1 → 5 | IPv6_QoS | 200 | 250 |
| 1 → 6 | IPv6_QoS | 200 | 250 |
| 1 → 7 | IPv6_QoS | 200 | 250 |
| 4 → 1 | IPv6_QoS | 200 | 250 |
| 5 → 1 | IPv6_QoS | 200 | 250 |
| 6 → 1 | IPv6_QoS | 200 | 250 |
| 7 → 1 | IPv6_QoS | 200 | 250 |
| 2 → 4 | CAN | 225 | 20 |
| 2 → 5 | CAN | 225 | 20 |
| 2 → 6 | CAN | 225 | 20 |
| 2 → 7 | CAN | 225 | 20 |
| 4 → 2 | CAN | 175 | 20 |
| 5 → 2 | CAN | 175 | 20 |
| 6 → 2 | CAN | 175 | 20 |
| 7 → 2 | CAN | 175 | 20 |
| 3 → 4 | DSS | 250 | 20 |
| 3 → 5 | DSS | 250 | 20 |
| 3 → 6 | DSS | 250 | 20 |
| 3 → 7 | DSS | 250 | 20 |
| 4 → 3 | DSS | 150 | 20 |
| 5 → 3 | DSS | 150 | 20 |
| 6 → 3 | DSS | 150 | 20 |
| 7 → 3 | DSS | 150 | 20 |

rzystania łączy, użyte zasoby sieciowe, liczba skoków (*hop count*) oraz opóźnienie transmisji. Wyniki przedstawiono dla wszystkich typów RI osobno oraz w formie wyników globalnych dla całej sieci. Naturalnym jest przyjęcie założenia, że poszczególne modele zachowują się najlepiej pod względem przyjętej funkcji celu. Stąd też np. Model 1 minimalizujący wykorzystanie przepustowości daje w istocie najmniejszą wartość wykorzystania zasobów łączy.

**Tablica 2.** Wyniki dla procedur wymiarowania zasobów i sieci zdefiniowanej na rysunku 1

| | | Model 1 | Model 2 | Model 3 |
|---|---|---|---|---|
| Przepustowość [Mbit/s] | IPv6_QoS | 7200 | 7250 | 7200 |
| | CAN | 12200 | 12200 | 12700 |
| | DSS | 8300 | 8750 | 8300 |
| | **Suma** | 27700 | 28200 | 28200 |
| Zasoby węzłowe [Mflops] | IPv6_QoS | 16000 | 16125 | 16000 |
| | CAN | 29275 | 29437 | 30775 |
| | DSS | 19950 | 21450 | 20100 |
| | **Suma** | 65225 | 67012 | 66875 |
| Liczba skoków | IPv6_QoS | 2.25 | 2.75 | 2.25 |
| | CAN | 3.75 | 4.12 | 3.87 |
| | DSS | 2.62 | 3.00 | 2.62 |
| | **Średnia** | 2.87 | 3.29 | 2.92 |
| Opóźnienie transmisji [ms] | IPv6_QoS | 10.50 | 12.75 | 10.50 |
| | CAN | 16.12 | 16.37 | 15.87 |
| | DSS | 11.35 | 13.25 | 11.75 |
| | **Średnia** | 12.66 | 14.12 | 12.71 |
| Całkowita przepustowość rezydualna (niezaalokowana) [Mbit/s] | | 16300 | 15800 | 15800 |

W Modelu 2 wprowadzono dodatkowe ograniczenia na wykorzystanie mocy obliczeniowej węzła. Model można łatwo rozszerzyć, aby uwzględniał kolejne elementy związane z węzłami, takie jak pamięć masowa czy też pamięć RAM. Takie dodatkowe ograniczenia mogą wpłynąć na wybór tras przez odrzucenie tras najkrótszych. Ponieważ warunki (ograniczenia) dotyczące zasobów węzłowych mogą być niezależne od zasobów łączy, powoduje to wprowadzenie dodatkowego stopnia swobody przy poszukiwaniu rozwiązania. W Modelu 3 wprowadzono dodatkowe ograniczenia związane z maksymalnymi dopuszczalnymi opóźnieniami transmisji dla każdego strumienia w relacji od krańca do krańca (*end-to-end*), obliczanymi jako wartość średnia opóźnienia przy przesyłaniu ruchu przez wszystkie łącza tworzące ścieżkę transmisyjną. Jednakże należy zauważyć, że ograniczenie wartości opóźnienia transmisji dla niektórych strumieni (żądań) może prowadzić do wyboru ścieżek, które nie są najkrótsze. Tłumaczy to uzyskanie

całkowitej wartości średniej opóźnienia dla Modelu 3, która jest wyższa niż dla Modelu 1.

## 4 Podsumowanie

Główne problemy do rozwiązania w obszarze wymiarowania to definicja pojemności sieci (zasobów transportowych), definicja pojemności węzła sieci, rozpoznanie zależności między pojemnością węzła fizycznego a pojemnością węzłów wirtualnych, metodologia dla wymiarowania sieci, która uwzględniałaby zależność między wymaganą przepustowością łącza wirtualnego a pojemnością węzła wirtualnego. Zdefiniowany moduł wymiarowania współpracuje ściśle z systemem zarządzania i zapewnia rozdział zasobów pomiędzy poszczególne RI z uwzględnieniem zmian zapotrzebowania oraz sytuacji awaryjnych. Obecnie są prowadzone prace implementacyjne realizujące rozdział zasobów sieciowych i węzłowych, a także bezpośrednią konfigurację parametrów platform sprzętowych.

## Literatura

1. W. Burakowski, H. Tarasiuk, A. Bęben, *Architektura Systemu IIP*, Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź, 14-16 września 2011.
2. W. Burakowski et al., *The IIP System: Architecture, Parallel Internets, Virtualization and Applications*, Future Network & Mobile Summit 2011, Warszawa, 15-17 czerwca 2011.
3. W. Burakowski, H. Tarasiuk, A. Bęben, *Future Internet Architecture Based on Virtualization and Co-existence of Different Data and Control Planes*, Fifth Future Internet Cluster Workshop focused on Future Network Architectures, Warszawa, 14 czerwca 2011.
4. M. Bourguiba, K. Haddadou, G. Pujolle, *Evaluating and Enhancing Xen-based Virtual Routers to Support Real-time Applications*, Proc. of IEEE CCNC 2010, Las Vegas, 9-12 January 2010.
5. J. Granat i in. *Zarządzanie i wymiarowanie zasobów w L1/L2 Systemu IIP*, Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Warszawa, 14-16 września 2011.
6. A. Fischer et al., *ALEVIN - A Framework to Develop, Compare, and Analyze Virtual Network Embedding Algorithms*, WowKiVS 2011.
7. Y. Zhu, M. Ammar, *Algorithms for Assigning Substrate Network Resources to Virtual Network Components*, Proc. of IEEE INFOCOM' 06, pp. 1-12, 2006.
8. N. Chowdhury, M. Rahman, R. Boutaba, *Virtual network embedding with coordinated node and link mapping*, Proc. of IEEE INFOCOM' 09, pp. 783-791, 2009.
9. J. Lischka, H. Karl, *A virtual network mapping algorithm based on subgraph isomorphism detection*, Proc. of ACM SIGCOMM VISA, 2009.
10. I. Houidi, W. Louati, D. Zeghlache, *A distributed virtual network mapping algorithm*, Proc. of IEEE ICC, pp. 5634-5640, 2008.
11. J. F. Botero et al. *Optimal mapping of virtual networks with hidden hops*, Telecommunication Systems, pp. 1-10, 2011.
12. J. Gozdecki i in. *A Flexible Provisioning Module for Optimizing Utilization of Resources for the Future Internet IIP Initiative*, Networks 2012, 15th International Telecommunications Strategy and Planning Symposium, 15-18.10.2012, Rzym.

# Równoległy Internet IPv6 QoS: system sygnalizacji i scenariusze aplikacyjne

Halina Tarasiuk[1], Jan Rogowski[1], Paweł Świątek[2], Krzysztof Brzostowski[2], Jarosław Drapała[2], Krzysztof Juszczyszyn[2], Cezary Mazurek[3], Bartosz Lewandowski[3], Arkadiusz Radziuk[3], Marcin Wrzos[3], Dariusz Walczak[3], Marek Natkaniec[4], Janusz Gozdecki[4], Marek Sikora[4]

[1] Politechnika Warszawska
[2] Politechnika Wrocławska
[3] Poznańskie Centrum Superkomputerowo-Sieciowe
[4] AGH Akademia Górniczo-Hutnicza w Krakowie

**Streszczenie** W artykule przedstawiono opis systemu sygnalizacji w Równoległym Internecie (RI) IPv6 QoS. System sygnalizacji odgrywa główną rolę w obsłudze żądań zasobów dla nowych połączeń i pozwala na sterowanie ich przyjmowaniem, aby zapewnić ścisłe gwarancje QoS dla przekazu pakietów przez sieć. Jako aplikacje testowe, w systemie badane są tzw. aplikacje Internetu Rzeczy, m.in. aplikacje e-zdrowie, zarządzanie energią, bezpieczeństwo publiczne oraz zdalne nauczanie. W artykule przedstawiamy typowe scenariusze aplikacyjne oraz przykładowe wyniki testów dla warstw usługowej i transportowej proponowanego systemu sygnalizacji.

## 1 Wprowadzenie

W artykule przedstawiamy propozycję realizacji systemu sygnalizacji dla Równoległego Internetu (RI) IPv6 QoS. Głównym założeniem dla RI IPv6 QoS jest obsługa połączeń z gwarancją jakości przekazu pakietów dla wybranych aplikacji testowych. Wśród aplikacji testowych wyróżniamy nowe aplikacje Internetu Rzeczy (Internet of Things – IoT) [5], takie jak e-zdrowie, monitorowanie i bezpieczeństwo publiczne, zarządzanie energią, oraz aplikacje przeznaczone do zdalnego nauczania: aplikacja zdalnego nauczania, która ma wbudowany system wideo-konferencji w jakości HD (High Definition) i aplikacja umożliwiająca zdalne obliczenia na rozproszonych serwerach obliczeniowych.

Wymienione aplikacje testowe realizują połączenia, z których część może być z powodzeniem obsługiwana w obecnej sieci Internet typu best effort. Jednak większość połączeń, głównie związanych z przekazem obrazów, danych z sensorów, a także aplikacje z grupy e-zdrowie wymagają gwarancji na jakość przekazu pakietów przez sieć. Przykładowo, aplikacje e-zdrowie, bądź też aplikacje związane z monitorowaniem przestrzeni publicznej np. w czasie imprez masowych, realizują funkcje, które mogą być krytyczne dla zdrowia i życia ludzkiego. W celu zapewnienia jakości przekazu pakietów w sieci dla wybranych połączeń konieczne jest zastosowanie odpowiednich algorytmów przyjmowania nowych wywołań.

Dlatego też, po analizie wymagań dla wymienionych aplikacji testowych zostały zdefiniowane klasy usług koniec-koniec dla odpowiedniej obsługi pakietów z wybranych połączeń aplikacji. Zaś w ramach poszczególnych klas usług, zostały zdefiniowane odpowiednie mechanizmy QoS w węzłach sieci IPv6 QoS oraz algorytmy dla realizacji funkcji przyjmowania nowych wywołań. Realizacja tych funkcji wymaga odpowiedniego systemu sygnalizacji. System ten w ramach RI IPv6 QoS jest zbudowany zgodnie z rekomendacjamii ITU-T dla architektury NGN (Next Generation Networks) [1] i z zaleceniami IETF dla architektury DiffServ (Differentiated Services) [2]. Nowym elementem funkcjonalnym, który zaproponowano w ramach RI IPv6 QoS, w porównaniu do obecnych rekomendacji ITU-T, jest zestaw funkcji w warstwie usługowej architektury NGN dla nowych typów aplikacji z grupy aplikacji Internetu Rzeczy. W projekcie zaproponowano funkcje warstwy usługowej specyficzne dla omawianych grup aplikacji testowych. Ich realizacja zostanie omówiona w dalszej części artykułu. Należy zauważyć, iż obecne rekomendacje ITU-T [1] dla warstwy usługowej odnoszą się do grup aplikacji multimedialnych takich jak IPTV, czy VoIP z rozwiązaniem IMS. Aplikacje multimedialne stosują jako protokół sygnalizacyjny w warstwie usługowej protokół SIP, którego mechanizmy nie muszą być konieczne dla realizacji takich aplikacji jak np. aplikacje Internetu Rzeczy.

W artykule przedstawiono wstępne wyniki testów systemu sygnalizacji. Wyniki obejmują czasy zestawienia i rozłączenia sesji połączeń.

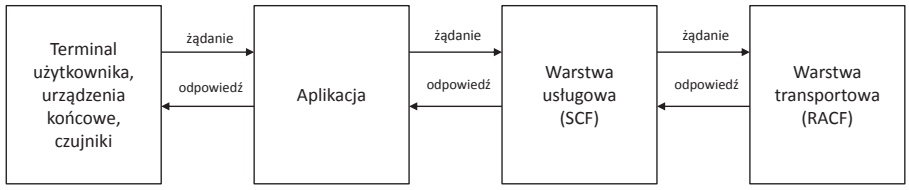
Układ artykułu jest następujący. Rozdział 2 przedstawia opis systemu sygnalizacji. Rozdziały 3–6 dotyczą scenariuszy aplikacyjnych dla aplikacji testowych sieci RI IPv6 QoS. W rozdziale 7 są prezentowane wstępne wyniki testów dla systemu sygnalizacji. Rozdział 8 stanowi podsumowanie.

## 2 System sygnalizacji

Rys. 1 przedstawia główne elementy systemu sygnalizacji zaproponowanego dla sieci RI IPv6 QoS z przykładową wymianą wiadomości sygnalizacyjnych. Głównymi elementami systemu są: terminale i urządzania końcowe, które sygnalizują do sieci żądania zasobów dla poszczególnych połączeń; warstwa związana z systemem aplikacyjnym – aplikacja; warstwa usługowa oraz warstwa transportowa. W dalszej części rozdziału zostaną omówione elementy funkcjonalne systemu sygnalizacji dla obu warstw: usługowej i transportowej.

### 2.1 Warstwa usługowa

Podstawowym celem sygnalizacji w warstwie usługowej jest: konfiguracja modułów rozproszonej aplikacji biorących udział w dostarczaniu żądanej funkcjonalności, negocjacja odpowiednich wartości parametrów komunikacyjnych oraz wyznaczenie i rezerwacja zasobów komunikacyjnych i obliczeniowych niezbędnych do wykonania żądanego scenariusza aplikacyjnego przy zagwarantowaniu odpowiedniej jakości usług.




**Rysunek 1.** Elementy systemu sygnalizacji z przykładową wymianą wiadomości sygnalizacyjnych dla sieci IPv6 QoS. SCF – Service Control Function – Funkcja warstwy usługowej, RACF – Resource and Admission Control Function – Funkcja sterowania zasobami oraz przyjmowania nowych wywołań

Celem przeprowadzenia negocjacji jest ustanowienie pomiędzy modułami aplikacji połączenia koniec-koniec z żądanymi wartościami parametrów QoS. Negocjacje polegające na wymianie odpowiednich komunikatów rozpoczynane są przez moduł aplikacyjny (usługę), który będzie wysyłał dane. Negocjacje realizowane są pomiędzy każdą parą usług pomiędzy którymi zachodzić będzie bezpośrednia komunikacja. W celu umożliwienia negocjacji pomiędzy parą dowolnych usług zaproponowano dwuetapowy protokół sygnalizacji umożliwiający definiowanie dowolnych parametrów specyficznych dla aplikacji, których wartości będą ustalane w drodze negocjacji.

### 2.2 Warstwa transportowa

Warstwa transportowa składa się z modułów funkcjonalnych systemu sygnalizacji oraz urządzeń sieciowych. Architektura NGN definiuje podział tej warstwy na funkcje sterowania warstwy transportowej (Transport Control Functions – TCF) oraz funkcje warstwy transportowej (Transport Functions – TF). Jako funkcje sterowania warstwy transportowej w systemie IPv6 QoS definiujemy funkcje sterowania zasobami i przyjmowania nowych wywołań (RACF), które są realizowane przez dwa moduły funkcjonalne: PD-FE (Policy Decision Functional Entity) i TRC-FE (Transport Resource Control Functional Entity). Natomiast za realizację funkcji warstwy transportowej odpowiadają moduły typu PE-FE (Policy Enforcement Functional Entity).

RACF jest odpowiedzialny za utrzymywanie informacji o zasobach sieciowych oraz wykonywanie algorytmów przyjmowania nowych wywołań. Główną funkcją modułu PD-FE jest komunikowanie się z modułem SCF i podejmowanie decyzji dotyczących żądań otrzymanych od SCF. Zgodnie z rekomendacjami ITU-T oraz IETF, każdy moduł PD-FE posiada interfejsy do komunikacji z modułami TRC-FE oraz PE-FE (odpowiednio Rt i Rw). Do komunikacji z innymi modułami PD-FE służy interfejs Rd. Możemy wyróżnić dwa rodzaje modułów PD-FE: sieci szkieletowej oraz sieci dostępowych. Moduł PD-FE sieci szkieletowej jest modułem centralnym i tylko ten moduł posiada interfejs do warstwy usługowej SCF (Rs). W tym module podejmowana jest decyzja o przyjęciu lub odrzuceniu

zgłoszenia. Decyzja ta podejmowana jest na podstawie mechanizmów przyjmowania nowych wywołań oraz decyzji pozostałych modułów PD-FE biorących udział w realizacji żądania QoS. Moduł PD-FE bierze udział we wszystkich scenariuszach sygnalizacji żądań obsługiwanych przez system sygnalizacji RI IPv6 QoS, takich jak: rezerwacja zasobów, zwalnianie zasobów, zmiana zasobów sieci dostępowej, czy przerwanie połączeń.

Moduł TRC-FE realizuje funkcje przyjmowania nowych wywołań do poszczególnych klas usług na podstawie informacji o zasobach dostępnych w węzłach brzegowych sieci. Jego głównym zadaniem jest podjęcie decyzji czy żądanie otrzymane od PD-FE może być zrealizowane. W tym celu moduł TRC-FE utrzymuje bazę dostępnych zasobów sieciowych dedykowanych dla poszczególnych klas usług w ramach sieci wirtualnych. Zasoby te są uaktualniane w wyniku działania modułu TRC-FE, np. przyjmowanie bądź rozłączanie połączeń, a także informacje od punktów dostępowych (Access Point – AP) w sieciach dostępowych. W tym module utrzymywana jest również lista aktywnych połączeń realizowanych przez system sygnalizacji IPv6 QoS. Moduł TRC-FE może również zainicjować rozłączenie niektórych z aktywnych połączeń po otrzymaniu informacji o zmniejszeniu dostępnych zasobów sieciowych (np. w sieci bezprzewodowej).

Moduł PE-FE odpowiada za komunikację z urządzeniem sieciowym w celu ustawienia parametrów klasyfikacji pakietów, monitorowania lub kształtowania strumieni pakietów w węzłach brzegowych sieci dla nowego połączenia.

Szczegóły związane ze schematami wymiany wiadomości w warstwie transportowej oraz formaty wiadomości zostały omówione w [3].

## 3 Scenariusze aplikacyjne – e-zdrowie

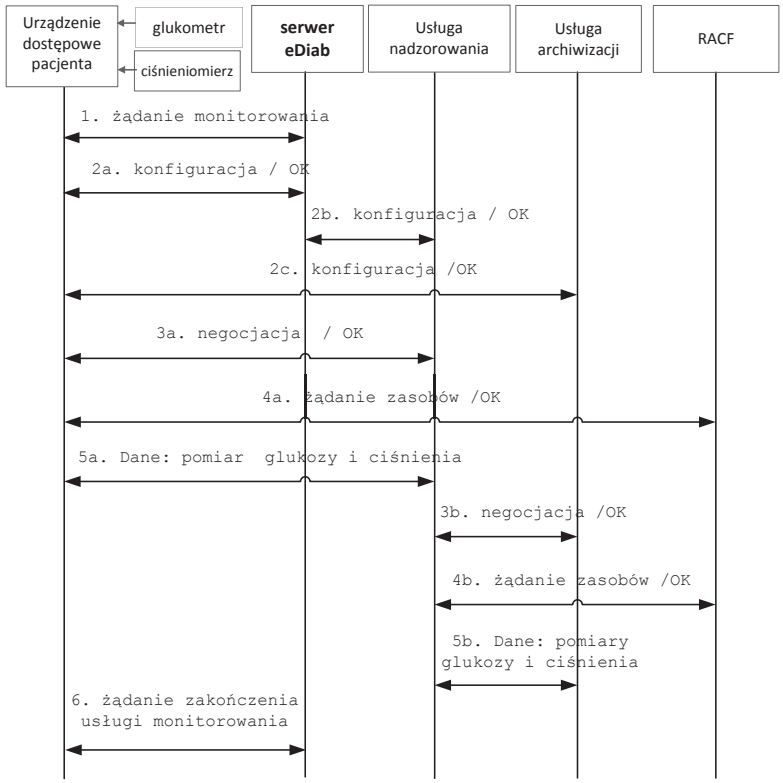
W rozdziale przedstawimy scenariusze aplikacyjne dla grupy aplikacji testowych e-zdrowie: eDiab, SmartFit oraz ASTMA.

### 3.1 eDiab

Podstawowym zadaniem aplikacji eDiab jest wspomaganie procesu zarządzania danymi pomiarowymi przez osobę chorą na cukrzycę typu II. Użytkownik aplikacji uzyskuje dostęp do określonych usług komunikacyjnych i obliczeniowych.

Wykonanie usługi rozpoczyna się od (Rys. 2) wygenerowania żądania monitorowania (1). Konfigurowane zostają połączenia: – pomiędzy smartfonem użytkownika a serwerem eDiab (2a), który udostępnia usługi, – pomiędzy serwerem eDiab a usługami nadzorowania (2b) i archiwizacji (2c). Następnie negocjowane są parametry połączeń (3a, 3b) związane z jakością obsługi w sieci RI IPv6 QoS.

Pomiary parametrów fizjologicznych wykonywane są za pomocą glukometru i ciśnieniomierza i przesyłane – z wykorzystaniem protokołu Bluetooth – urządzenia dostępowego klienta. Żądania zasobów sieci (4a, 4b) są realizowane za pośrednictwem serwera RACF. Od tego momentu dane pomiarowe i wyniki ich



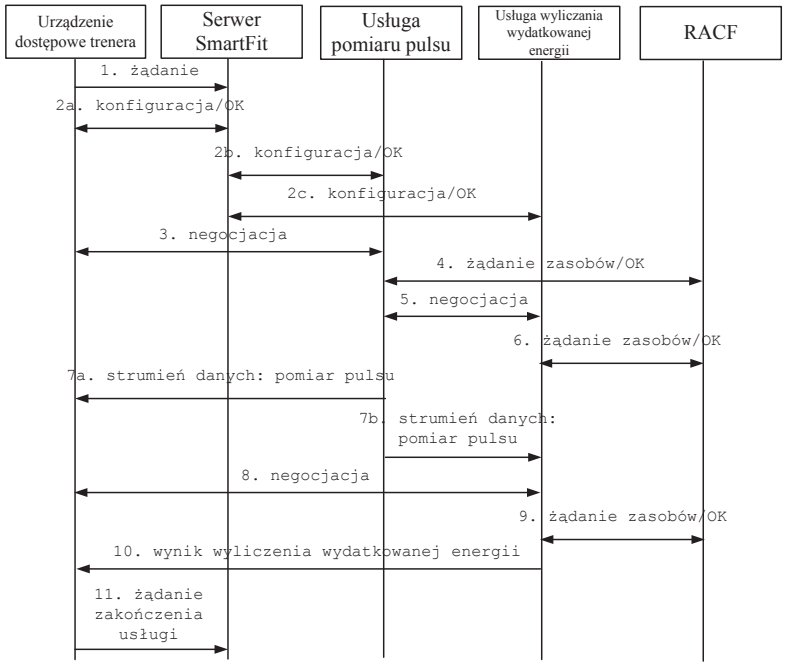

**Rysunek 2.** Podstawowy scenariusz monitorowania osoby chorej na cukrzycę typu II

przetwarzania (np. klasyfikacji pomiarów przez usługę nadzorowania) są przesyłane w ramach połączeń odpowiednich klas usług sieciowych. Zakres zadań, jakie może wykonywać aplikacja, zależy od zestawu dostępnych usług.

### 3.2 SmartFit

Aplikacja SmartFit jest narzędziem przeznaczonym do wspomagania planowania i nadzorowania treningu wytrzymałościowego oraz technicznego sportowców z wykorzystaniem technologii bezprzewodowych. Dostępne funkcjonalności omawianego rozwiązania zostały uzyskane poprzez implementację mechanizmów pozwalających na bezprzewodowy pomiar parametrów fizjologicznych i kinematycznych oraz opracowanie metod przetwarzania danych.

Jednym z elementów treningu wytrzymałościowego jest monitorowanie jego intensywności. Uzyskane wyniki pozwalają ocenić na bieżąco zaangażowanie sportowca podczas treningu jak również zweryfikować, czy wykonane zostały zaplanowane zadania.




**Rysunek 3.** Scenariusz usługi monitorowania treningu wytrzymałościowego dla aplikacji SmartFit

Na Rys. 3 przedstawiono przykładowy scenariusz usługi monitorowania treningu wytrzymałościowego dla aplikacji SmartFit. Wykonanie usługi odbywa się w kilku etapach, wśród których można wymienić: żądanie wykonania i zakończenia usługi, konfigurację, negocjację oraz transfer danych. Wykonanie usługi monitorowania treningu wytrzymałościowego rozpoczyna się od wysłania przez użytkownika żądania wykonania usługi (1). Następny etap związany jest z konfiguracją usług wchodzących w skład omawianego scenariusza (2a, 2b i 2c).

W kolejnym etapie następuje negocjacja pomiędzy usługami, które będą wymieniały pomiędzy sobą dane. Wysyłane jest również odpowiednie żądanie rezerwacji zasobów do serwera RACF (3 i 4 oraz 5 i 6). Po pomyślnym zarezerwowaniu zasobów następuje etap transferu danych (7a i 7b). Operacja negocjacji oraz żądania zasobów może być wykonywana wielokrotnie w ramach danego scenariusza (8 i 9). Ostatnim etapem wykonania usługi jest wysłanie przez użytkownika żądania zakończenia wykonywania usługi (11).

### 3.3 ASTMA

Aplikacja zdalnego monitorowania stanu zdrowia pacjenta chorego na astmę (w skrócie aplikacja ASTMA) obejmuje kilka scenariuszy aplikacyjnych, w których

należy zapewnić wymaganą jakość transmisji danych. Główne elementy aplikacji ASTMA to: urządzenia do monitorowania po stronie pacjenta (rytmu serca, świstów w płucach, efektywności płuc), główny serwer e-zdrowie z aplikacją ASTMA (BD Astma) i centralną bazą danych, stanowisko pracy lekarza dyżurującego i bazę danych zanieczyszczeń. Do komunikacji pomiędzy elementami aplikacji ASTMA wybrano sieć RI IPv6 QoS. Spośród scenariuszy aplikacyjnych najważniejsze to:

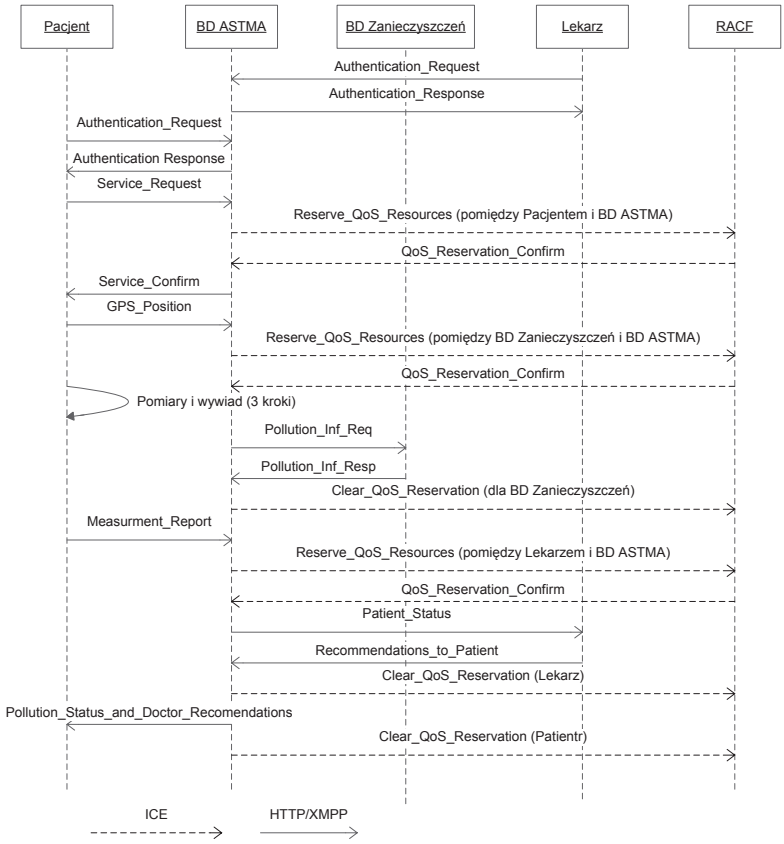
- okresowe, codzienne testy poranne i wieczorne stanu zdrowia pacjenta;
- testy stanu zdrowia pacjenta wykonywane na żądanie — zwykle wykonywane w razie pogorszenia stanu zdrowia pacjenta;
- monitorowanie stanu zdrowia pacjenta w trybie ciągłym;
- telekonsultacja pacjenta z lekarzem dyżurującym.

W skład każdego ze scenariuszy aplikacyjnych wchodzi scenariusz sygnalizacyjny, którego zadaniem jest rezerwacja odpowiednich zasobów w sieci RI IPv6 QoS, tak aby komunikacja pomiędzy elementami aplikacji ASTMA odbywała się z wymaganą jakością, niezbędną do prawidłowego działania.

Na Rys. 4 przedstawiono scenariusz sygnalizacyjny dla codziennych, porannych i wieczornych testów stanu zdrowia pacjenta chorego na astmę. Sygnalizacja w tym scenariuszu jest podzielona na sygnalizację pomiędzy elementami aplikacji ASTMA za pomocą protokołów HTTP (Hypertext Transfer Protocol) i XMPP (Extensible Messaging and Presence Protocol), oraz na sygnalizację pomiędzy głównym serwerem aplikacji ASTMA, a serwerem RACF. Scenariusz ten można podzielić na trzy etapy, które się wzajemnie przeplatają. Pierwszy etap obejmuje komunikację lekarza dyżurującego z głównym serwerem ASTMA. Na początku lekarz dyżurujący rejestruje się w systemie i oczekuje na zlecenia. Gdy system posiada zadanie do wykonania przez lekarza dyżurującego, rezerwuje wymagane zasoby sieciowe w sieci RI IPv6 QoS (komunikacja pomiędzy modułem BD ASTMA i serwerem RACF na Rys. 4) i powiadamia lekarza, który zdalnie wykonuje swoje zadanie w systemie. Po wykonaniu zleconego zadania przez lekarza zasoby sieciowe są zwalniane. Drugi etap obejmuje komunikację pomiędzy pacjentem i głównym serwerem ASTMA.

W pierwszej fazie drugiego etapu pacjent rejestruje się w systemie ASTMA i żąda dostępu do usługi codziennych testów. System ASTMA rezerwuje zasoby sieciowe pomiędzy pacjentem i głównym serwerem. Następnie przesyłane są dane pomiarowe od pacjenta na główny serwer aplikacji. Na serwerze następuje analiza danych pomiarowych i wynik tej analizy z ewentualnymi zaleceniami lekarza dyżurującego jest przesyłany do pacjenta. Po odesłaniu wyników do pacjenta zasoby sieciowe są zwalniane. Trzeci etap obejmuje komunikację pomiędzy głównym serwerem ASTMA i bazą danych zanieczyszczeń (BD Zanieczyszczeń).

W tym etapie po uzyskaniu informacji o położeniu geograficznym pacjenta, system rezerwuje zasoby sieciowe potrzebne do transmisji informacji o zanieczyszczeniu w lokalizacji pacjenta i pobiera te informacje. Następnie zwalnia zarezerwowane zasoby. Scenariusze sygnalizacyjne dla testów stanu zdrowia pacjenta wykonywanych na żądanie i testów wykonywanych w trybie ciągłym są

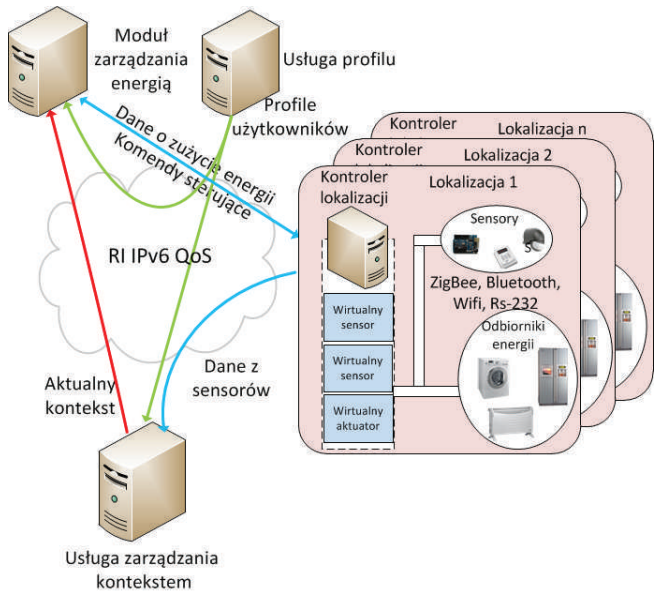



**Rysunek 4.** Scenariusz sygnalizacyjny dla codziennych, porannych i wieczornych testów stanu zdrowia pacjenta chorego na astmę

uproszczoną wersją scenariusza opisanego powyżej. Ponadto, scenariusz sygnalizacyjny obejmujący telekonsultację pacjenta z lekarzem dyżurującym obejmuje standardową sygnalizację dla usługi VoIP za pomocą protokołu SIP pomiędzy terminalem lekarza, terminalem pacjenta i serwerem SIP na głównym serwerze aplikacji ASTMA, oraz komunikację pomiędzy głównym serwerem aplikacji ASTMA i serwerem RACF w systemie sygnalizacji sieci RI IPv6 QoS.

## 4 Scenariusze aplikacyjne - zarządzanie energią

Aplikacja HomeNetEnergy (Rys. 5) [4] realizuje scenariusze z obszaru Internetu Rzeczy [5] związane z monitorowaniem zużycia oraz inteligentnym zarządzaniem energią elektryczną w rozproszonych lokalizacjach – z wykorzystaniem informacji kontekstowej oraz profili użytkowników. Aplikacja łączy w sobie zalety podejścia




**Rysunek 5.** Elementy składowe aplikacji HomeNetEnergy oraz ich wzajemna komunikacja

Internetu Rzeczy z architekturą zorientowaną na zdarzenia – w celu umożliwienia analizy zużycia energii przez urządzenia elektryczne w czasie zbliżonym do rzeczywistego, ręcznego zarządzania odbiornikami energii przez użytkowników, oraz inteligentnego zarządzania odbiornikami przez samą aplikację. Zaproponowane scenariusze aplikacyjne w ogólności dotyczą dwóch obszarów powszechnie znanych pod pojęciem komunikacja typu maszyna-maszyna (Machine-2-Machine, M2M) [6], tj. akwizycji danych z rozproszonych czujników, oraz bezpośredniej interakcji z urządzeniami wykonawczymi.

W aplikacji zdefiniowano trzy scenariusze pokrywające swoim zakresem wyżej wymienioną funkcjonalność. Są to:

- scenariusz zdalnego monitorowania zużycia energii w rozproszonych lokalizacjach;
- scenariusz ręcznego sterowania odbiornikami;
- scenariusz inteligentnego zarządzania zużyciem energii z wykorzystaniem informacji kontekstowej oraz profili użytkowników.

W scenariuszu zdalnego monitorowania zużycia energii w rozproszonych lokalizacjach aplikacja obsługuje dwa tryby monitorowania. Pierwszy z nich pozwala na śledzenie aktualnego chwilowego zużycia energii. Zdarzenia związane z pobieraniem danych są wyzwalane przez świadome działanie użytkownika, który może odpytywać kolejne urządzenia o ich aktualny poziom konsumpcji. Drugi tryb umożliwia automatyczne zbieranie danych i ich zapis w repozytorium danych

aplikacji w celu ich późniejszej wizualizacji lub analizy zgodnie z zadanym przez użytkownika interwałem czasowym, na przykład co 10 minut. W oparciu o dane taryfowe, aplikacja umożliwia analizę kosztów zużycia energii dla poszczególnych urządzeń w wybranych przedziałach czasu, jak również porównanie kosztów w zależności od dostępnych na rynku taryf dystrybutorów energii.

Scenariusz ręcznego sterowania odbiornikami prezentuje możliwości aplikacji w zakresie zdalnej kontroli nad elementami wykonawczymi. W tym scenariuszu przepływ komunikatów sterujących następuje od aplikacji do urządzeń fizycznych, a funkcjonalność aplikacji pozwala użytkownikom na załączanie i odłączanie zasilania elektrycznego poszczególnym odbiornikom.

Ostatni scenariusz, inteligentnego zarządzania zużyciem energii z wykorzystaniem informacji kontekstowej i profilu użytkownika, łączy w sobie zarówno akwizycję danych z czujników, jak również sterowanie urządzeniami wykonawczymi. Przetworzone przez usługę kontekstu dane z czujników otoczenia, wraz z preferencjami użytkowników są dostarczane do modułu zarządzania energią, który generuje na ich podstawie komendy sterujące, wysyłane do urządzeń wykonawczych. Reguły przetwarzania są definiowane przez użytkownika aplikacji i mogą być zmieniane w dowolnym momencie.

Przedstawione powyżej scenariusze (Rys. 5) wykorzystują dwa protokoły do komunikacji z otoczeniem: HTTP oraz XMPP. Protokół HTTP jest wykorzystywany do przesyłania danych, które nie są krytyczne dla działania aplikacji, np. dane o chwilowym zużyciu wysyłane na żądanie użytkownika w pierwszym scenariuszu. Z kolei protokół XMPP jest wykorzystywany do przesyłania danych krytycznych dla prawidłowego funkcjonowania aplikacji, które wymagają dostarczenia z gwarancją zachowania jakości obsługi. Są to między innymi komendy sterujące oraz dane z sensorów wymagające małych opóźnień dla przekazu przez sieć. Dla tego typu komunikacji aplikacja wykorzystuje Równoległy Internet IPv6 QoS, rezerwując niezbędne zasoby sieci.

## 5 Scenariusze aplikacyjne - bezpieczeństwo publiczne

MobiWatch jest platformą integracyjną dla rozproszonych systemów bezpieczeństwa publicznego i monitoringu [7]. Platforma ta integruje dane pochodzące z systemów monitoringu wizyjnego, zarówno z systemów telewizji przemysłowej jak i z nowoczesnych cyfrowych systemów monitoringu, z danymi pochodzącymi z sieci sensorowych. Multimedialne strumienie danych są przesyłane i przetwarzane przez system MobiWatch, a następnie wzbogacane o dodatkowe informacje dotyczące wybranych obiektów, ludzi, czy zdarzeń zachodzących w obszarze monitorowania, oraz o odpowiednie metadane. W tym celu wykorzystany jest standard MPEG-7. System umożliwia również dołączanie informacji kontekstowych lub pochodzących z zewnętrznych źródeł danych. Dodatkowe informacje mogą być uzyskane z wykorzystaniem usług analizy audio i wideo, dzięki sensorom zintegrowanym z systemem oraz z dodatkowych baz danych. Platforma MobiWatch wykorzystuje techniki automatycznego przetwarzania zdarzeń oraz paradygmat architektury zorientowanej na usługi [8]. Dzięki tym cechom stosun-

kowo bezproblemowe jest rozszerzanie systemu o zestaw nowych komponentów przetwarzających dane multimedialne, ich opisy oraz o różnego rodzaju aplikacje klienckie. Dzięki przetwarzaniu zdarzeń z wykorzystaniem techniki CEP (Complex Event Processing) możliwe jest monitorowanie na bieżąco stanu systemu oraz wykrywanie sytuacji podejrzanych i krytycznych minimalizując czas reakcji [9]. MobiWatch składa się z luźno połączonych modułów aplikacyjnych, które podzielić można na 3 grupy: warstwę przesyłania multimediów, warstwę usług oraz warstwę przetwarzania zdarzeń. Elastyczne połączenie modułów zrealizowane jest przy pomocy tzw. mechanizmu publish-subscribe [10], w którym wszystkie elementy systemu pełnią rolę producenta treści, odbiorcy lub obie te funkcje równocześnie. Każdy producent treści wysyła do węzła systemu pewien zbiór danych. Każdy odbiorca treści zapisuje się do węzła i definiuje, jakiego typu danymi jest zainteresowany. Kiedy pojawiają się one w węźle, zostają natychmiast przesłane do odpowiednich odbiorców.

W systemie MobiWatch wyróżnić można dwie klasy zastosowania. Pierwsza związana jest z pracą osoby odpowiedzialnej za wdrażanie systemu i rozszerzanie go o nowe urządzenia i usługi. W skład tej klasy scenariuszy wchodzi scenariusz związany z dodaniem do systemu nowej kamery i nowego sensora (np. czujki ruchu) oraz scenariusz konfiguracji tych urządzeń w systemie korzystając z odpowiedniego interfejsu użytkownika. Innym scenariuszem jest przekierowanie strumienia multimedialnego z serwera strumieniowego do aplikacji analizującej jego zawartość, np. do aplikacji wykrywania twarzy w obrazie. Druga klasa scenariuszy związana jest z pracą operatora systemu w centrum monitoringu. W skład tej grupy scenariuszy wchodzi podgląd na żywo obrazu z kamer, wyszukiwanie odpowiednich materiałów archiwalnych przy pomocy słów kluczowych znajdujących się w ich opisach multimedialnych oraz podgląd tych materiałów. Analizując wszystkie przypadki użycia związane z użytkownikiem w systemie MobiWatch wyróżnić można 4 typy połączeń realizowanych przez sieć. Każdy z tych modeli jest odwzorowywany do innej klasy usług oferowanej przez sieć IPv6 QoS. Wyróżnić można następujące scenariusze:

- pozyskiwanie strumieni multimedialnych na żywo;
- odtwarzanie strumieni multimedialnych na żywo;
- odtwarzanie materiałów archiwalnych,
- przesyłanie informacji o zdarzeniach.

W scenariuszach sieciowych związanych z przekazywaniem treści multimedialnych aplikacja MobiWatch wykorzystuje protokół RTP (Real Time Protocol). Komunikacja maszyna-maszyna (M2M) pomiędzy komponentami aplikacyjnymi oraz sensorami odbywa się z wykorzystaniem protokołu XMPP. W ten sposób przesyłane są między innymi wszystkie dane związane z wykryciem zdarzeń lub z przekazaniem informacji od sensorów. Interfejsy użytkownika zrealizowane są jako strony WWW.

## 6 Scenariusze aplikacyjne – zdalne nauczanie

Prototyp aplikacji Online Lab jest wirtualnym laboratorium obliczeniowym, pozwalającym definiować zadania obliczeniowe, publikować je dla innych użytkowników oraz wykonywać zdalnie, wykorzystując funkcje optymalizacji czasu odpowiedzi. Dostęp do aplikacji odbywa się przez graficzny interfejs WWW, co pozwala na jej wykorzystanie w zdalnym nauczaniu.

Scenariusz realizacji usługi obliczeniowej w Online Lab przedstawiono na Rys. 6. Użytkownik korzystając z wirtualnego pulpitu dostępnego za pomocą przeglądarki, edytuje kod oraz ma możliwość dołączenia do niego plików z danymi. Zdefiniowane w ten sposób żądanie realizacji obliczeń (1) przesyłane jest do modułu zarządzającego Online Lab (OL-Core Server). Następuje oszacowanie wymaganych zasobów obliczeniowych oraz kontrola stanu usług i wybór usługi obliczeniowej przez moduł równoważenia obciążeń (Load Balancer – LB, odpowiednio 4-6, 7-8 i 9). LB przekazuje decyzję o wyborze usługi do OL-Core (10), który dokonuje rezerwacji zasobów dla transferu danych i kodu z wykorzystaniem serwera RACF (11 i 12). Dane i kod są przesyłane do usługi obliczeniowej (13) a wynik obliczeń do OL-Core (14), który dokonuje formatowania wyników prezentowanych następnie użytkownikowi (15). OL-Core oraz LB gromadzą i przetwarzają dane dotyczące wszystkich żądań, które są następnie analizowane w celu rezerwacji zasobów w sieci.

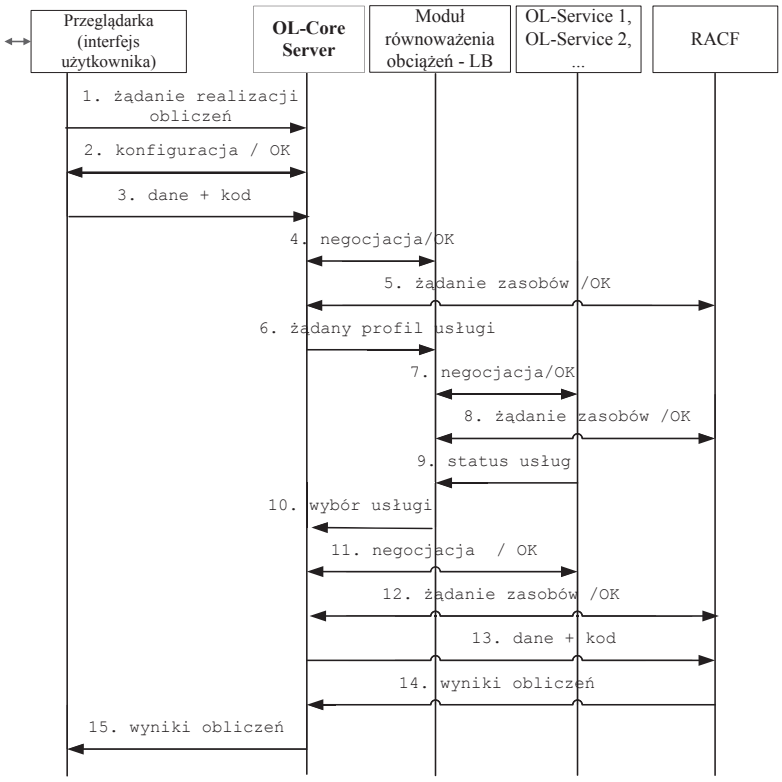
## 7 Wstępne wyniki badań

W rozdziale przedstawiono wstępne wyniki badań dla systemu sygnalizacji w warstwach: usługowej i transportowej.

### 7.1 Warstwa usługowa

Celem badań warstwy usługowej była weryfikacja efektywności zaproponowanego dwuetapowego schematu sygnalizacji realizowanego w ramach Uniwersalnej Platformy Komunikacyjnej (UPK). Efektywność sygnalizacji mierzona była jako czas nawiązywania grafu połączeń pomiędzy rozproszonymi usługami realizującymi poszczególne scenariusze użycia aplikacji. Badania polegały na pomiarze czasu nawiązania wszystkich połączeń koniec-koniec pomiędzy usługami realizującymi żądany scenariusz użycia aplikacji. Przykładowy scenariusz dla aplikacji SmartFit przedstawiony na Rys. 7 wymaga zestawienia trzech połączeń koniec-koniec:

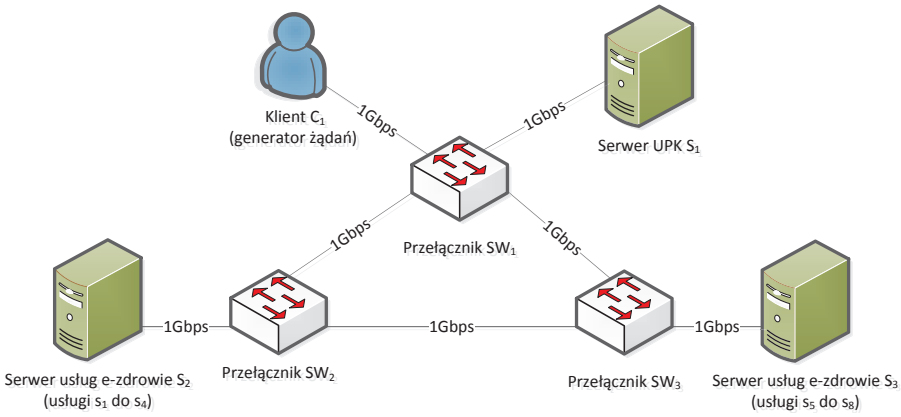
1. pomiędzy urządzeniem dostępowym klienta, a usługą pomiaru pulsu – transmisja danych z czujników;
2. pomiędzy usługą pomiaru pulsu, a usługą wyliczania wydatkowanej energii – transmisja przetworzonych danych z czujników;
3. pomiędzy usługą wyliczania wydatkowanej energii, a urządzeniem klienta - transmisja wyników przetwarzania danych, w tym podjętych decyzji dotyczących treningu.




**Rysunek 6.** Scenariusz realizacji usług obliczeniowych w Online Lab

Na Rys. 7 przedstawiono topologię testową składającą się z trzech przełączników warstwy trzeciej SW1 – SW3 (Juniper EX4200), trzech serwerów usług S1 – S3 (Intel Core i7 960, 12GB RAM) oraz aplikacji klienckiej C1 pełniącej rolę generatora żądań zestawienia scenariuszy aplikacyjnych. Na serwerze S1 uruchomiony był serwer UPK, na serwerach usług e-zdrowie S2 i S3 uruchomiono po cztery usługi aplikacyjne. Klient C1 wysyłał do UPK żądania zestawienia scenariuszy aplikacyjnych składających się z usług oferowanych przez serwery S2 i S3. Rozmiar scenariuszy (mierzony jako liczba połączeń pomiędzy usługami wchodzącymi w skład scenariusza) oraz intensywność napływu żądań były losowane zgodnie z rozkładem równomiernym.

Badania efektywności systemu sygnalizacji w warstwie usługowej polegały na pomiarze czasu ustanowienia wszystkich połączeń dla poszczególnych żądań. Czas był mierzony od momentu wysłania żądania przez klienta do momentu potwierdzenia przez serwer UPK [11] utworzenia wszystkich połączeń. Wyniki przeprowadzonych badań przedstawione są na Rys. 8.




**Rysunek 7.** Topologia dla testów systemu sygnalizacji w warstwie usługowej

Podczas badań generowane i realizowane były żądania zestawienia grafów połączeń o różnych rozmiarach (tj. o różnej liczbie połączeń koniec-koniec). Żądania były generowane z różną intensywnością. Przeprowadzono 5 serii eksperymentów, w których badano wpływ rozmiaru generowanych żądań oraz intensywności ich napływu na czas ich realizacji. Serie eksperymentów 1–4 składały się z pięciu uruchomień każda. W każdej serii generowano żądania o zadanym rozmiarze $k = 1, \ldots, 4$ (w pierwszej serii o rozmiarze $k = 1$, w drugiej o rozmiarze $k = 2$, itd.) natomiast w kolejnych uruchomieniach w danej serii zwiększano intensywność generowania żądań ($\lambda = 1, \ldots, 5$). Odstępy czasu pomiędzy kolejnymi żądaniami były losowane z rozkładu wykładniczego z parametrem $\lambda$. I tak w pierwszym uruchomieniu w serii pierwszej żądania zestawienia jednokrawędziowych grafów połączeń były generowane ze średnią intensywnością $\lambda = 1$ żądanie na sekundę. W piątym uruchomieniu serii czwartej generowano żądania czterokrawędziowe z intensywnością $\lambda = 5$ żądań na sekundę.

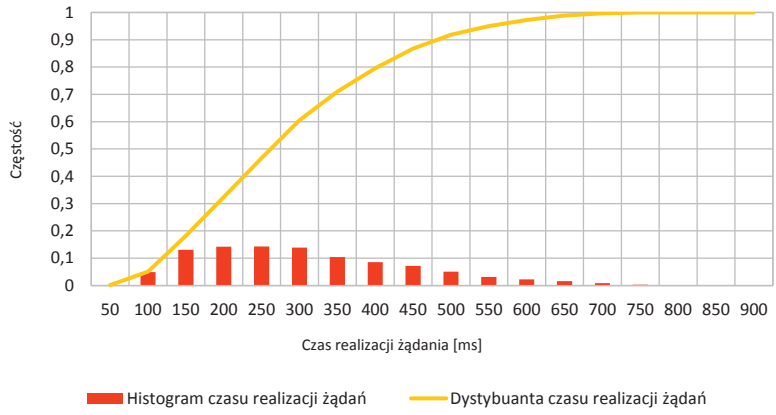
Piąta seria składała się z jednego uruchomienia, w którym rozmiary żądań były losowe ($k = 1, \ldots, 4$), a intensywność ich napływu zmieniała się w czasie ($\lambda = 1, \ldots, 5$). Każde uruchomienie w każdej z serii polegało na wygenerowaniu 20000 żądań. W Tab. 1 przedstawiono wynik przeprowadzonych badań dla każdego uruchomienia z każdej serii.

Na Rys. 8 przedstawiono histogram i dystrybuantę empiryczną dla przykładowego uruchomienia w eksperymencie z serii piątej, w którym zarówno rozmiary żądań, jak i intensywność ich napływu zmieniały się losowo w trakcie trwania eksperymentu. Sytuacja ta jest zbliżona do rzeczywistego uruchomienia systemu sygnalizacji w warstwie usługowej i można uznać ją za reprezentatywną.

Z przeprowadzonych badań wynika, że czasy zestawiania grafów połączeń koniec-koniec dla poszczególnych przypadków użycia aplikacji rosną liniowo wraz ze wzrostem liczby połączeń w scenariuszach oraz intensywnością napływu żądań. Warto zauważyć, że w żadnym z rozpatrywanych scenariuszy aplikacyjnych

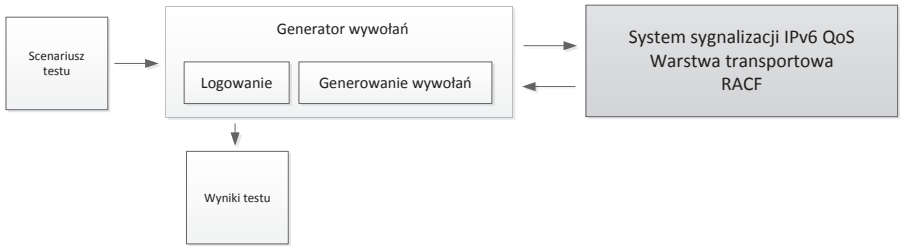
| Nr serii | Rozmiar żądania $k$ | Średni czas realizacji żądania [ms] | 95-percentyl czasu realizacji żądania [ms] |
|---|---|---|---|
| 1 | 1 | 234.54 | 411.23 |
| 2 | 2 | 260.39 | 472.74 |
| 3 | 3 | 376.45 | 547.41 |
| 4 | 4 | 463.39 | 678.12 |
| 5 | 1,2,3,4 | 308.69 | 552.17 |

**Tablica 1.** Wartość średnia oraz 95-percentyl czasów realizacji żądań zestawienia grafów połączeń koniec-koniec




**Rysunek 8.** Histogram i dystrybuanta czasów realizacji żądań w eksperymencie serii piątej – losowe rozmiary i intensywności napływu żądań

liczba połączeń niezbędnych do zestawienia całej usługi nie przekracza czterech. Dodatkowo intensywność napływu żądań $\lambda = 5$ oznacza, że co sekundę do systemu zgłaszać się będzie pięciu nowych użytkowników, co jest mało prawdopodobne. Dla największych rozpatrywanych scenariuszy i przy największym obciążeniu systemu średni czas realizacji żądań wynosił 463 ms, a 95-percentyl 678 ms. Przy zbliżonym do rzeczywistych warunków uruchomieniu eksperymentu, w którym rozmiary żądań i intensywność ich napływu zmienia się w czasie, średni czas realizacji usługi wynosił 308 ms, a 95-percentyl 552 ms. Analiza wyników badań wskazuje, że dla rozpatrywanych przypadków użycia, które są reprezentatywne dla rzeczywistych uruchomień systemu, czasy realizacji żądań są o rząd wielkości niższe od zaleceń ITU-T dla np. aplikacji VoIP [12].




**Rysunek 9.** Ogólny scenariusz badania

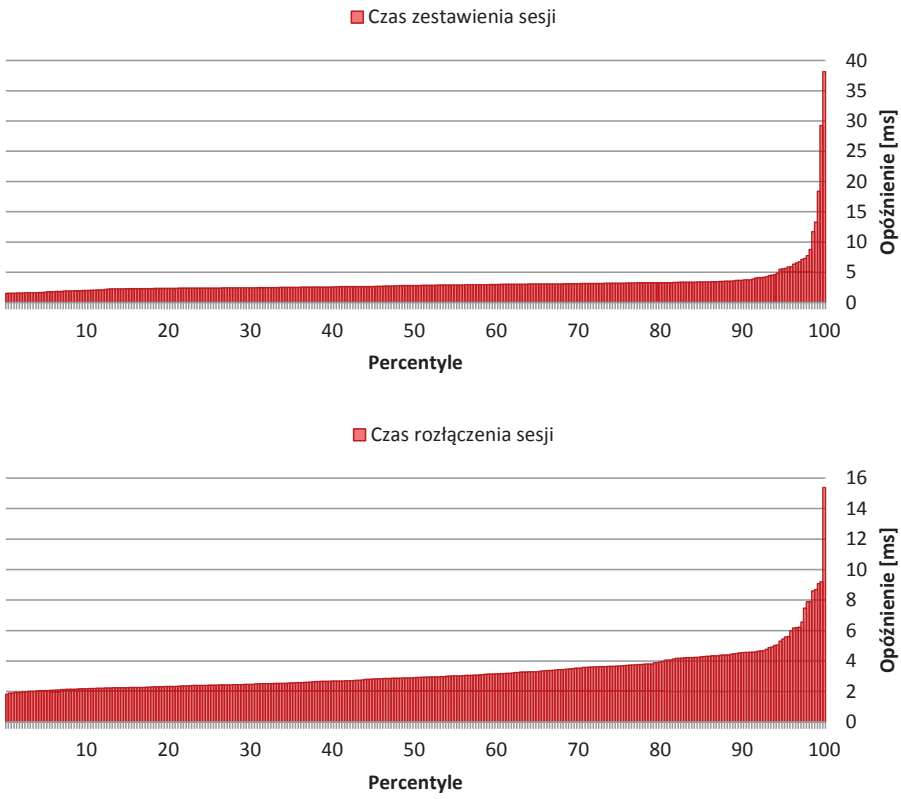
### 7.2 Warstwa transportowa

Celem przeprowadzonych testów była ocena wydajności systemu sygnalizacji IPv6 QoS w warstwie transportowej. Dokładniej, zbadanie wydajności implementacji algorytmów dla modułów sygnalizacyjnych RACF oraz komunikacji między nimi. W ramach testów nie zbadano komunikacji systemu sygnalizacji z urządzeniami sieciowymi, będzie to przedmiotem dalszych eksperymentów. W ramach testów badanymi metrykami były: czas zestawienia sesji oraz czas rozłączenia sesji. Czas ten, był mierzony od chwili wywołania funkcji generującej żądanie do chwili odebrania odpowiedzi. Ponieważ czas zestawienia sesji ma wpływ na jakość usługi odbieraną przez użytkownika, jako wartości graniczne przyjęto te z zalecenia ITU-T, podobnie jak w [12]. Zgodnie z zaleceniami, dla połączeń krajowych średni czas zestawienia sesji nie powinien przekraczać 5 s, natomiast 95-percentyl nie powinien być większy niż 8 s. System sygnalizacji został uruchomiony na jednej maszynie wirtualnej. Na tej samej maszynie wirtualnej uruchomiony był generator wywołań. Rys. 9 przedstawia ogólny scenariusz badania.

Scenariusz zakładał wygenerowanie 300 zgłoszeń zestawienia sesji, a następnie 300 zgłoszeń rozłączenia sesji. Zgłoszenia, zarówno zestawienia jak i rozłączenia sesji, były generowane z intensywnością 100 zgłoszeń na sekundę w stałych odstępach czasowych. Test został powtórzony 5 razy.

Zgodnie ze scenariuszem testowym każdy z 5 przeprowadzonych testów zakończył się pozytywnym zestawieniem oraz rozłączeniem 300 sesji. W Tab. 2 przedstawiono średnie czasy zestawienia i rozłączenia sesji z poziomem ufności 95%. Jak można zauważyć, wartość średnia jest znacznie niższa niż wymagania przedstawione w [12] ($< 5s$).

Dla czasu zestawienia sesji 95-percentyl wynosi 5,6 ms, natomiast dla czasu rozłączenia sesji 4,5 ms. Zarówno 95-percentyl dla czasu zestawienia sesji, jak i dla czasu rozłączenia sesji są mniejsze od przyjętych w zaleceniach ($< 8s$). Maksymalny czas zestawienia sesji był dużo większy od wartości średniej co jest efektem obciążenia procesora innymi zadaniami, nie zaś efektem przeciążenia systemu sygnalizacji IPv6 QoS. Wartości średnie czasu zestawienia sesji we wszystkich 5 testach są na poziomie 3 ms co przy odstępach między wywołaniami wynoszącymi około 10 ms pozwala stwierdzić poprawność implementacji




**Rysunek 10.** Charakterystyka opóźnienia wynikającego z czasu zestawienia sesji

algorytmów i struktur danych. Przy napływie zgłoszeń rzędu 100 zgłoszeń/s system sygnalizacji wykazuje dużą wydajność i czas odpowiedzi nie przekraczający wymaganego poziomu.

| Wynik pomiaru | Wartość minimalna | Wartość średnia | Wartość maksymalna |
|---|---|---|---|
| Czas zestawienia sesji [ms] | 1.48 | $3.20 \pm 0.52$ | 52.66 |
| Czas rozłączenia sesji [ms] | 1.34 | $3.33 \pm 0.23$ | 33.58 |

**Tablica 2.** Wyniki badań czasu zestawienia i rozłączenia sesji

Na Rys. 10 przedstawiono wykresy percentyli czasu zestawienia sesji oraz czasu rozłączenia sesji.

## 8 Podsumowanie

W artykule zaprezentowano system sygnalizacji dla Równoległego Internetu IPv6 QoS oraz scenariusze dla aplikacji testowych implementowanych w ramach projektu Inżynieria Internetu Przyszłości i uruchamianych w sieci RI IPv6 QoS. Wstępne wyniki testów przeprowadzonych dla prototypowej implementacji systemu sygnalizacji dla warstw usługowej i transportowej systemu RI IPv6 QoS pokazują, iż czasy zestawienia sesji spełniają wymagania stawiane czasom obsługi np. dla aplikacji typu VoIP. Czasy referencyjne dla sygnalizacji aplikacji VoIP stanowią w przeprowadzonych badaniach punkt odniesienia dla aplikacji Internetu Rzeczy, ponieważ obecnie brak jest jeszcze rekomendacji dla tego typu aplikacji.

## Literatura

1. Rec. Y.: ITU-T, General overview of NGN, ITU Telecommunication Standardisation Sector, December 2004
2. Blake, S. i inni: An Architecture for Differentiated Services, Internet RFC 2475, 10/1998
3. Tarasiuk, H., Rogowski, J.: On the Signaling System in the IPv6 QoS Parallel Internet, Proc. of CSNDSP 2012, Poznań 2012
4. Lewandowski, B., Mazurek, C., Radziuk, A.: Aplikacja inteligentnego zarządzania energią w środowisku domowym jako usługa Internetu Przyszłości, Materiały Konferencja i3 2010, Wrocław 2010
5. Tan, L., Wang, N.: Future Internet: The Internet of Things, Proc. Of 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), str. 376 – 380, 2010
6. Wu, G. i inni: "M2M": From mobile to embedded internet, Communications Magazine, IEEE, 49 (4), str. 36–43, 2011
7. Mierzwiński, D., Walczak, D., Wolski, M., Wrzos, M.: Video and Sensor Data Integration in a Service-Oriented Surveillance System, Scalable Computing: Practice and Experience (12), 2011
8. Michelson, B. Event-Driven Architecture Overview: Event-Driven SOA Is Just Part of the EDA Story, `http://www.omg.org/soa/Uploaded%20Docs/EDA/bda2-2-06cc.pdf`, Dostęp: luty 2006
9. Luckham, D.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems, Addison-Wesley Professional, 2002
10. Eugster, P.T., et al., The many faces of publish/subscribe, Tech. Rep. DSC ID:2000104, EPFL 2001
11. Świątek, P. et al., Supporting content, context and user awareness in Future Internet applications, Lecture Notes in Computer Science (7281), str. 154–165, 2012
12. Tarasiuk, H. i inni: Performance Evaluation of Signaling in the IP QoS System, Journalof Telecommunications and Information Technology (3), str. 12–20, 2011

# Implementacja płaszczyzny danych i płaszczyzny sterowania dla Równoległego Internetu IPv6 QoS

Piotr Wiśniewski[1], Halina Tarasiuk[1], Mariusz Rawski[1], Iwo Olszewski[2], Łukasz Ogrodowczyk[2], Sławomir Hanczewski[3], Adam Kaliszan[3], Agnieszka Brachman[4], Krzysztof Gierłowski[5]

[1] Instytut Telekomunikacji, Politechnika Warszawska
[2] Poznańskie Centrum Superkomputerowo-Sieciowe
[3] Instytut Elektroniki i Telekomunikacji, Politechnika Poznańska
[4] Instytut Informatyki, Politechnika Śląska
[5] Katedra Teleinformatyki, Politechnika Gdańska

**Streszczenie** W artykule przedstawiono aspekty związane z implementacją węzłów Równoległego Internetu (RI) IPv6 QoS na platformach umożliwiających wirtualizację. RI IPv6 QoS jest oparty na architekturze DiffServ i NGN. Zgodnie z architekturą DiffServ, w sieci IP wyróżniamy węzły brzegowe oraz szkieletowe, które wyposażone są w mechanizmy Quality of Service (QoS) wspierające gwarancje QoS dla przekazu pakietów w sieci. Implementacja węzłów IPv6 QoS w środowisku umożliwiającym wirtualizację polega na rozdzieleniu płaszczyzny przekazu danych od płaszczyzny sterowania oraz stworzeniu odpowiednich interfejsów między wydzielonymi płaszczyznami z warstwami L1/L2 Systemu IIP. W artykule przedstawiono szczegóły implementacji warstwy przekazu danych i sterowania w środowisku wirtualnym Systemu IIP.

## 1 Wprowadzenie

Technika wirtualizacji stanowi obecnie jedną z podstawowych technik Internetu Przyszłości [1]. W szczególności wirtualizacja węzłów i łączy sieci umożliwia stworzenie równolegle działających sieci (Równoległych Internetów [2]), które mogą być zbudowane w oparciu o różne stosy protokołów, nie tylko TCP/IP. Technika wirtualizacji węzłów i łączy pozwala, aby jeden węzeł infrastruktury fizycznej sieci był współdzielony między kilka węzłów wirtualnych oraz, aby jedno łącze fizyczne sieci było współdzielone między kilka łączy wirtualnych [2]. Artykuł jest poświęcony zagadnieniom związanym z realizacją węzła i łącza wirtualnego dla Równoległego Internetu (RI) IPv6 QoS. W szczególności, propozycji nowego podejścia do implementacji płaszczyzny danych i sterowania dla węzłów sieci IP QoS. W podejściu tym, węzeł RI IPv6 QoS współdzieli z innymi Równoległymi Internetami zasoby węzła infrastruktury fizycznej sieci i stanowi tzw. węzeł wirtualny. Współdzielone jest także łącze fizyczne. Ponadto, implementacja płaszczyzny danych i sterowania dla każdego węzła sieci IPv6 QoS

jest podzielona na dwa odrębne moduły funkcjonalne, z których każdy może być umieszczony na innym urządzeniu. Takie podejście daje nowe możliwości dla realizacji funkcji sterowania i zarządzania w sieci. Dodatkowo, dla RI IPv6 QoS definiujemy możliwość tworzenia sieci wirtualnych, które pozwalają na rozdzielenie tablic rutingu. Dzięki tej funkcjonalności operatorzy sieci wirtualnych uzyskują możliwość zarządzania swoją siecią wirtualną.

Szczegóły związane z architekturą dla RI IPv6 QoS zostały przedstawione w [3]. Architektura ta jest oparta na architekturze DiffServ (Differentiated Services) [4] i NGN (Next Generation Networks) [5]. Zgodnie z architekturą DiffServ węzły sieci spełniają wymagania związane z mechanizmami umożliwiającymi różnicowanie klas usług w sieci oraz ścisłymi gwarancjami jakości przekazu pakietów przez sieć dla wybranych połączeń. Dlatego też, w warstwie transportowej architektury mają zastosowanie zarówno mechanizmy QoS dla przekazu danych, typu PHB (Per Hop Behaviour), jak i mechanizmy systemu sygnalizacji, w tym realizacja funkcji przyjmowania nowych wywołań.

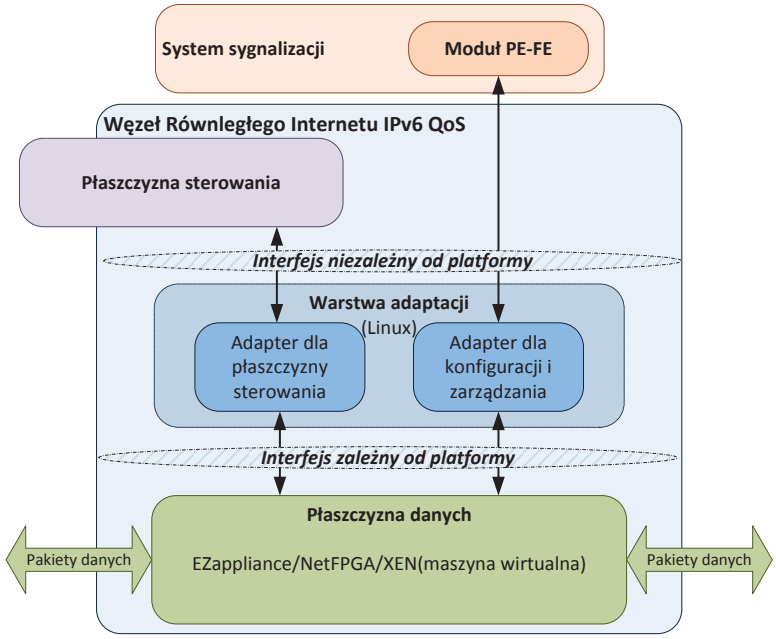
Mimo, iż obecnie sieci bazujące na architekturze DiffServ i NGN są wdrażane przez operatorów sieci Internet w oparciu o rutery IP QoS dostępne na rynku, to nowe podejście do implementacji tego typu sieci na infrastrukturze umożliwiającej wirtualizację wydaje się być niezbędnym krokiem, aby zintegrować sieć IP QoS z rozwiązaniami Internetu Przyszłości bazującymi na wirtualizacji, które są tworzone z myślą o nowych rozwiązaniach dla aplikacji i usług.

Wśród prac związanych z techniką wirtualizacji węzłów sieci, w ramach badań nad Internetem Przyszłości, możemy wyróżnić np. wyniki projektu 4WARD [6]. W projekcie tym, implementacja węzłów wirtualnych była przeprowadzona dla protokołu IPv4, zaś sieć IP realizowała jedynie obsługę pakietów na zasadzie best effort. Kolejnym podejściem do wirtualizacji węzłów, które wspiera rozdział płaszczyzny przekazu danych i sterowania w ramach jednego węzła jest rozwiązanie Openflow [7], jednakże platformy umożliwiające wirtualizację wybrane dla implementacji rozwiązań proponowanych w projekcie nie dysponowały interfejsami do komunikacji z narzędziem Openflow dla płaszczyzny danych.

Układ artykułu jest następujący. Rozdział 2. przedstawia model warstwy transportowej oraz model węzła RI IPv6 QoS. Rozdziały 3. i 4. prezentują aspekty implementacyjne odpowiednio dla płaszczyzny sterowania oraz płaszczyzny danych. Rozdział 5. jest poświecony współpracy RI IPv6QoS z siecią best effort. Rozdział 6. stanowi podsumowanie.

## 2 Model węzła sieci RI IPv6 QoS

Zgodnie z architekturą NGN, architektura RI IPv6 QoS jest podzielona na dwie główne warstwy: warstwę transportową oraz warstwę usługową. W artykule przedstawiamy podejście do implementacji węzłów sieci IP, które stanowią elementy warstwy transportowej architektury NGN. W szczególności, zgodnie z architekturą DiffServ [4], warstwa transportowa RI IPv6 QoS składa się z ruterów brzegowych (RB) i ruterów szkieletowych (RS).




**Rysunek 1.** Ogólny model węzła sieci RI IPv6 QoS

Zgodnie z założoną koncepcją płaszczyzna danych ruterów RI IPv6 QoS (RB i RS) jest zrealizowana w postaci węzłów wirtualnych na urządzeniach umożliwiających wirtualizację [2]. W szczególności, płaszczyzna danych RB została zaimplementowana na urządzeniach EZappliance [8], natomiast płaszczyzna danych RS została zaimplementowana na kartach NetFPGA [9] oraz w środowisku XEN [10]. Płaszczyzna sterowania została zrealizowana na oddzielnych węzłach (szczegóły w rozdziale 3 oraz w [11]). Rys. 1 przedstawia ogólny model węzła sieci RI IPv6 QoS, który jest niezależny od platformy sprzętowej. W celu zapewnienia jednolitego sposobu komunikacji pomiędzy heterogeniczną płaszczyzną danych i płaszczyzną sterowania oraz systemem sygnalizacji została wprowadzona warstwa adaptacji. Adapter dla konfiguracji i zarządzania zapewnia uniwersalny interfejs dla modułu systemu sygnalizacji PE-FE (Policy Enforcement Functional Entity) [12], natomiast adapter dla płaszczyzny sterowania zapewnia standardowy sposób przekazu informacji o rutingu (np. tablic rutingu).

## 3 Płaszczyzna sterowania

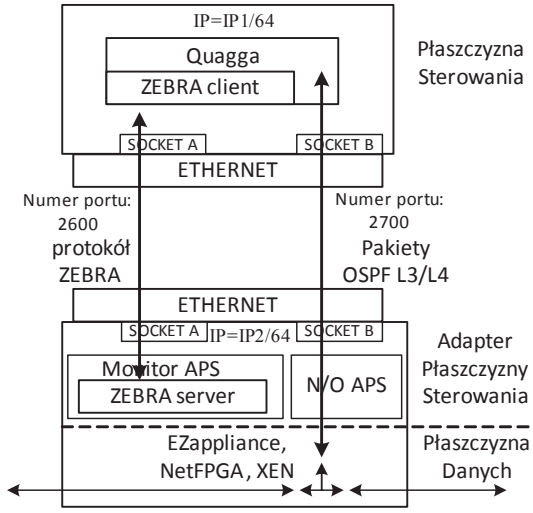
W węźle sieci RI IPv6 QoS, funkcje płaszczyzny sterowania (PS) są realizowane przez ruter programowy Quagga [13]. Ruter ten został w całości napisany

w języku C, dzięki czemu charakteryzuje się bardzo dużą stabilnością pracy. Punktem wyjściowym tworzenia PS dla sieci RI IPv6 QoS jest Quagga w wersji 0.99.18 (wydanie z dnia 23.03.2011) Do realizacji funkcji rutingu wykorzystywany jest protokół OSPFv3. Wyjściowa wersja rutera Quagga została dostosowana do wymogów budowanej sieci RI IPv6 QoS. W pierwszym etapie implementacji ruter Quagga został zmodyfikowany tak, by można było uruchamiać go na innej maszynie (fizycznej lub wirtualnej) niż urządzenie płaszczyzny danych (PD). Wyniesienie rutera Quagga na osobną maszynę umożliwiło uniezależnienie oprogramowania płaszczyzny sterowania od architektury urządzenia realizującego funkcjonalność płaszczyzny danych. W węzłach sieci RI IPv6 QoS (Rys. 2) są wykorzystywane trzy różne urządzenia płaszczyzny danych (NetFPGA, EZappliance, XEN) [16]. Każde z nich jest tak samo traktowane przez płaszczyznę sterowania. Jest to możliwe, gdyż komunikacja pomiędzy ruterem programowym a urządzeniem płaszczyzny danych jest realizowana poprzez dodatkowy moduł adapter płaszczyzny sterowania (APS). W drugim etapie implementacji została zapewniona komunikacja pomiędzy PS a urządzeniem PD, aby umożliwić wysyłanie i odbieranie informacji niezbędnych do wyznaczenia przez PS ścieżek w sieci oraz przesłanie tablic rutingu do urządzenia PD. W tym przypadku do komunikacji zastosowano zmodyfikowany na potrzeby projektu IIP protokół Zebra [13] (w adapterze APS funkcje te realizuje moduł Monitor APS). Drugim elementem, który został zaprojektowany jest przesyłanie pomiędzy ruterami programowymi, obsługującymi różne węzły, wiadomości protokołu OSPFv3 (zarówno na poziomie sieci IP, jak i na poziomie sieci wirtualnych). Odpowiednią komunikację uzyskano dzięki wykorzystaniu tunelowania "IP w IP"(moduł N/O APS). Równolegle z pracami implementacyjnymi zapewniającymi odpowiednią komunikację pomiędzy PS a APS prowadzone były prace nad kolejną modyfikacją rutera Quagga, dzięki której (zgodnie z przyjętą specyfikacją PS) w ramach jednego procesu rutingu OSPFv3 można niezależnie realizować funkcje rutingu dla wszystkich obsługiwanych sieci wirtualnych. W efekcie prac implementacyjnych, komunikacja pomiędzy płaszczyzną sterowania a urządzeniem płaszczyzny danych realizowana jest w sposób przedstawiony na Rys. 2.

Komunikacja pomiędzy poszczególnymi modułami APS a ruterem programowym jest realizowana za pomocą dwóch połączeń TCP (odpowiednie numery portów zaznaczone są na Rys. 2. Adresy IP, dzięki którym PS i PD wymieniają pomiędzy sobą informacje mają znaczenie lokalne i informacja o nich pozostaje w obrębie węzła sieci RI IPv6 QoS.

### 3.1 Protokół rutingu

Do realizacji funkcji rutingu w sieci RI IPv6 QoS został wybrany protokół OSPFv3. Zgodnie z przyjętymi założeniami protokół ten musi realizować funkcje rutingu dla sieci RI IPv6 QoS oraz wszystkich działających w jej ramach sieci wirtualnych (Rys. 3). Protokół OSPFv3 charakteryzuje się dużą stabilnością pracy, która jest wymagana w procesie wymiarowania sieci oraz gwarantuje jej poprawną pracę. Niewątpliwą zaletą protokołu OSPFv3 jest również jego skalowalność, dzięki której możliwa jest budowa dużych sieci. Na Rys. 3 została przed-

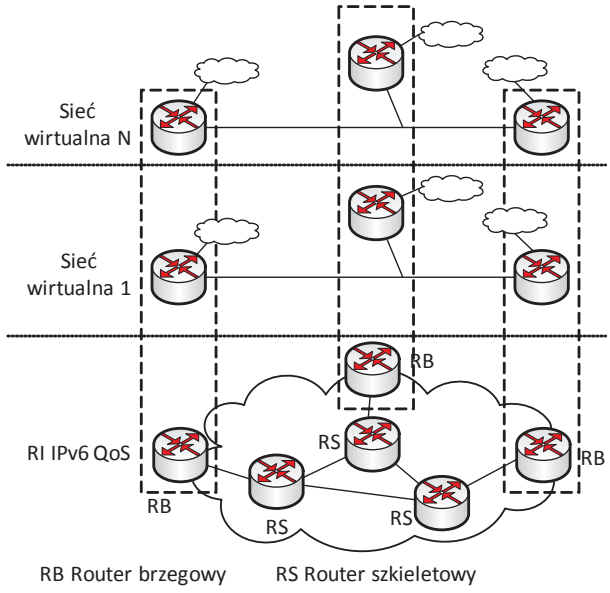



**Rysunek 2.** Komunikacja pomiędzy płaszczyzną sterowania a płaszczyzną danych. APS - adapter płaszczyzny sterowania; N/O - nadajnik/odbiornik

stawiona struktura sieci RI IPv6 QoS z punktu widzenia działania protokołu rutingu. Sieci wirtualne są implementowane powyżej warstwy sieci IP. Sieci te nie są świadome topologii sieci IPv6 QoS. Ponieważ sieci wirtualne działają niezależnie od siebie, dla każdej z nich konieczne jest uruchomienie oddzielnego procesu rutingu. Wspólnym elementem Równoległego Internetu IPv6 QoS oraz działających w jego ramach sieci wirtualnych są węzły brzegowe. Niezależnie od topologii sieci IPv6 QoS, z punktu widzenia sieci wirtualnych, rutery brzegowe połączone są w topologii gwiazdy. Funkcje węzłów szkieletowych sieci RI IPv6 QoS ograniczają się do przekazywania pakietów zgodnie z tablicami rutingu. Funkcjonalność węzłów brzegowych jest bardziej złożona, ponieważ z jednej strony biorą udział w procesie rutingu na poziomie sieci IP, z drugiej zaś zapewniają ruting na poziomie sieci wirtualnych. Aby maksymalnie uprościć i jednocześnie zoptymalizować działanie płaszczyzny sterowania dla ruterów brzegowych, przyjęto, że funkcje rutingu będą realizowane niezależnie w ramach pojedynczego procesu protokołu OSPFv3. Dzięki takiemu rozwiązaniu ograniczono zużycie zasobów koniecznych do budowy węzła sieci oraz uproszczono proces zarządzania węzłami.

## 4 Płaszczyzna danych

Płaszczyzna danych, jak wspomniano wcześniej, jest implementowana na trzech platformach. W dalszej części omówimy aspekty związane z implementacją me-

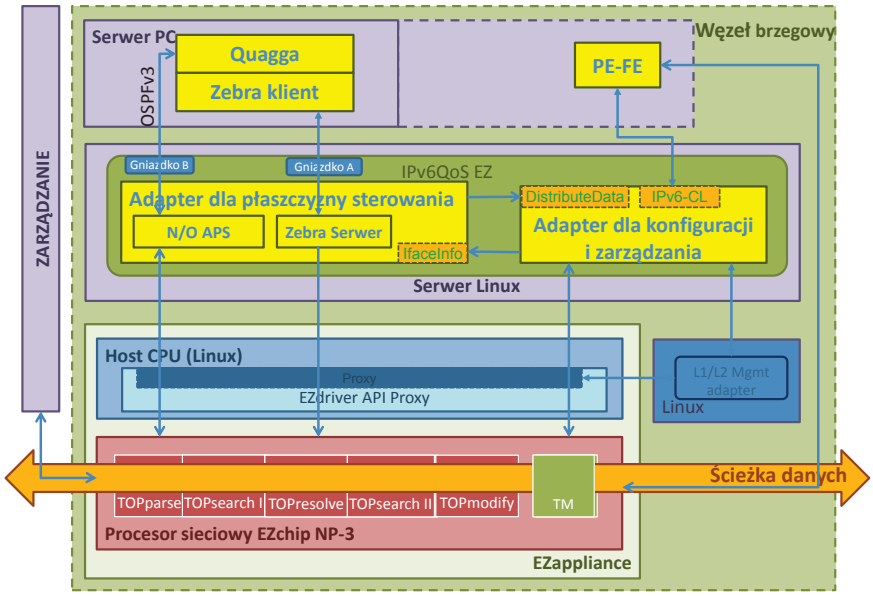



**Rysunek 3.** Ruting w sieci RI IPv6 QoS

chanizmów sieci RI IPv6 QoS na poszczególnych platformach: EZappliance, XEN i NetFPGA.

### 4.1 Węzeł EZappliance

Architekturę węzła brzegowego EZappliance dla RI IPv6 QoS przedstawia Rys. 4. Implementacja węzła RI IPv6 na platformie EZappliance składa się z czterech głównych elementów. Pierwszy to kod przygotowany w asemblerze procesorów TOP, który realizuje przetwarzanie pakietów (a właściwie ich nagłówków) w procesorze sieciowym EZchip NP-3. Odpowiada on za: klasyfikowanie pakietów, wszystkie decyzje rutingowe, a także realizuje mechanizm monitorowania i odrzucania pakietów (policing). Drugim elementem jest tzw. element zarządzania ruchem (Traffic Manager - TM ) realizujący następujące funkcje: kolejkowanie pakietów (queuing), szeregowanie pakietów (schedulling) oraz kształtowanie ruchu (shaping). Adapter, czyli program konfiguracyjno-sterującym stanowi trzeci z elementów węzła. Odpowiada on za konfigurację dwóch poprzednich elementów i zapewnia komunikację z płaszczyznami zarządzania i sterowania. Czwartą częścią implementacji węzła jest oprogramowanie płaszczyzny sterowania, które zostało opisane w rozdziale 3. Dwa ostatnie elementy są w implementacji prototypu uruchamiane na wydzielonym serwerze połączonym z urządzeniem EZappliance.




**Rysunek 4.** Architektura węzła brzegowego EZappliance

Węzeł EZappliance pełni rolę węzła brzegowego sieci IPv6 QoS, dlatego musi realizować wszystkie jego funkcje wynikające z architektury DiffServ [4] i NGN [5] oraz architektury RI IPv6 QoS [3]. I tak, w TOPparse, pierwszym ze specjalizowanych podprocesorów, nagłówki odebranego pakietu są analizowane i na ich podstawie tworzone są klucze struktur przeszukiwania oraz wiadomość dla kolejnych elementów TOP. Po fazie przygotowawczej procesor TOPsearch I przeszukuje odpowiednie struktury i przekazuje wyniki do TOPresolve, w którym podejmowane są na ich podstawie odpowiednie decyzje. W ten sposób jest realizowana większość funkcji węzła, czyli klasyfikacja pakietu do odpowiedniej wirtualnej sieci, przeszukanie tablicy rutingu danej sieci wirtualnej (VRF), aby poznać adres wyjściowego rutera brzegowego, a także przeszukanie tablicy rutingu sieci IP w celu określenia interfejsu wyjściowego. Jednocześnie realizowana jest funkcjonalność klasyfikacji połączenia (Multi Field Classifier - MF) i określenie odpowiednich mechanizmów QoS (w ramach TM) dla tego połączenia w ramach danej klasy usług. Ze względu na wirtualizację, port wyjściowy oraz numer kolejki w TM muszą zostać określone i przekazane niższej warstwie (poprzez odpowiedni rejestr) już w TOPresolve. Ostatnim etapem przetwarzania pakietu są modyfikacje realizowane w procesorze TOPmodify (w szczególności dodanie nagłówka tunelu "IP w IP") i przekazanie pakietu do niższej warstwy obsługi.

Ponieważ EZappliance pełni rolę węzła brzegowego sieci IPv6 QoS to posiada on funkcjonalność sterowania ruchem zarówno dla poszczególnych stru-

mieni ruchu jak i dla klas usług. Dla strumieni pakietów napływających z sieci dostępowych następuje ich klasyfikacja do poszczególnych połączeń (klasyfikator MF), po czym w module TM pakiety z poszczególnych strumieni mogą być kolejkowane (mechanizm drop-tail queue) i podlegać mechanizmowi kształtowania ruchu. W kolejnym etapie strumienie są agregowane do klas usług (klasyfikator BA - Behaviour Aggregate). Następnie pakiety należące do poszczególnych klas usług są kolejkowane (mechanizm drop-tail queue) i szeregowane (mechanizm Modified Deficit Round Robin).

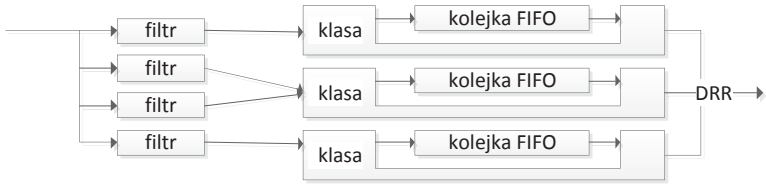
Adapter IPv6 QoS jest częścią węzła EZappliance. Moduł ten działa jako warstwa pośrednia pomiędzy płaszczyzną sterowania i płaszczyzną przekazu danych, a także zapewnia interfejs komunikacyjny z modułami systemu sygnalizacji warstwy transportowej architektury NGN. W rozwiązaniu prototypowym adapter IPv6 QoS jest realizowany jako maszyna wirtualna zainstalowana na serwerze Linux. W ramach adaptera wyróżniono dwa moduły programowe adapter dla płaszczyzny sterowania oraz adapter dla konfiguracji i zarządzania.

Adapter dla płaszczyzny sterowania składa się z dwóch modułów N/O APS (nadajnik/odbiornik adaptera płaszczyzny sterowania), który jest odpowiedzialny za obsługę protokołu rutingu OSPFv3 zarówno dla poziomu sieci IP jak i poziomu sieci wirtualnych, oraz moduł Zebra serwer do obsługi zmodyfikowanego protokołu Zebra. Z kolei, adapter dla konfiguracji i zarządzania zapewnia komunikację z modułem PE-FE systemu sygnalizacji. Oba omawiane moduły komunikują się ze sobą wykorzystując narzędzie Corba. W rozwiązaniu prototypowym zostały napisane w języku Python.

W ramach współpracy z płaszczyzną sterowania w adapterze IPv6 QoS został zaimplementowany moduł Zebra serwer będący częścią zmodyfikowanego (na potrzeby Równoległego Internetu IPv6 QoS) projektu Quagga. Główne zmiany w porównaniu z oryginalnym protokołem Zebra, na styku pomiędzy Zebra klient implementowanym na serwerze PC a Zebra serwer, polegają na dodaniu obsługi wirtualnych sieci, a także na wykorzystaniu funkcji API do konfiguracji struktur wyszukiwania (drzew binarnych) w procesorze NP3 urządzenia EZappliance do implementacji tablic rutingu.

Interfejs południowy adaptera wykorzystuje API dostarczone przez rozwiązanie EZchip do konfiguracji struktur wyszukiwania, oraz parametrów pracy elementu TM związanych z mechanizmami kształtowania ruchu, jak również umożliwia przesyłanie pakietów OSPFv3 pomiędzy Quaggą a procesorem NP3. Interfejs Proxy służący do komunikacji adaptera z procesorem NP3 został zaimplementowany w Host CPU - wydzielonej części urządzenia EZappliance, która bazuje na systemie Linux i umożliwia projektowanie wysokopoziomowych aplikacji. Komunikacja adaptera IPv6 QoS z Host CPU odbywa się z użyciem narzędzia Corba.

Interfejs IPv6-CL (IPv6-Command Line) umożliwia konfigurację węzła EZappliance. Odpowiednie polecenia z parametrami umożliwiają wprowadzenie m.in. adresów IPv6 interfejsów na płaszczyźnie danych, adresu IPv6 typu loopback węzła brzegowego, parametru RuterID, statycznych wpisów do tablic rutingu - wszystkich ustawień dokonuje się dla poziomu sieci IP oraz sieci wirtualnych.




**Rysunek 5.** Przykładowy schemat obsługi pakietów w module sterowania ruchem w węźle szkieletowym sieci IPv6 QoS na platformie XEN

W krótkiej skali czasu interfejs IPv6-CL umożliwia konfigurację mechanizmów QoS dla nowych połączeń przyjmowanych do obsługi i związaną z tym procesem konfigurację węzła.

### 4.2 Węzeł XEN

Węzeł IPv6 QoS w technologii XEN stanowi programową implementację węzła wirtualnego. Płaszczyzna danych realizowana przez węzeł XEN składa się z części wspólnej dla Systemu IIP (klasyfikator, algorytm szeregowania pakietów pierwszego poziomu) oraz części właściwej dla każdego z Równoległych Internetów. Funkcjonalność płaszczyzny danych dla poszczególnych Równoległych Internetów została zaimplementowana w maszynach wirtualnych dedykowanych dla każdego z nich. Maszyna wirtualna Równoległego Internetu IPv6 QoS pracuje w środowisku systemu operacyjnego Linux.

W Równoległym Internecie IPv6 QoS, węzeł XEN został zaprojektowany jako ruter szkieletowy. Płaszczyzna danych tego rutera realizuje następującą funkcjonalność: klasyfikacja pakietów na podstawie pola TC (Traffic Class) nagłówka IPv6 pakietu do zdefiniowanych klas usług,szeregowanie pakietów w ramach poszczególnych klas usług w celu zapewnienia podziału pasma przyjętego w procesie wymiarowania. Dla realizacji mechanizmów klasyfikacji i szeregowania pakietów zastosowano moduł sterowania ruchem (Traffic Control) [14] systemu operacyjnego Linux. Moduł ten składa się z kilku mechanizmów, do których należą m.in.: filtry, klasy, algorytmy obsługi kolejek. Schematyczny model obsługi pakietów zdefiniowany w module sterowania ruchem na potrzeby Równoległego Internetu IPv6 QoS przedstawia Rys. 5.

Klasyfikacja pakietów jest wykonywana na podstawie filtrów, które w zależności od wartości pola TC nagłówka IPv6 przydzielają pakiety do zdefiniowanych klas usług. Z każdą klasą usług związana jest kolejka, w której pakiety obsługiwane są zgodnie z dyscypliną FIFO (ang. First In, First Out). Wyjście kolejek jest obsługiwane zgodnie z algorytmem szeregowania pakietów DRR (ang. Deficit Round Robin) [15] lub PQ-DRR (ang. Priority Queue Deficit Round Robin).

### 4.3 Węzeł NetFPGA

Węzeł sieci szkieletowej RI IPv6 QoS został również zrealizowany na platformie NetFPGA. Platforma ta składa się z karty PCI wyposażonej w układ programowalny FPGA Virtex-II Pro 50 firmy Xilinx i czterech portów Ethernet oraz z oprogramowania i sterowników umożliwiających zarządzanie tą kartą [9]. Karta ta jest instalowana w komputerze klasy PC działającym pod kontrolą systemu Linux.

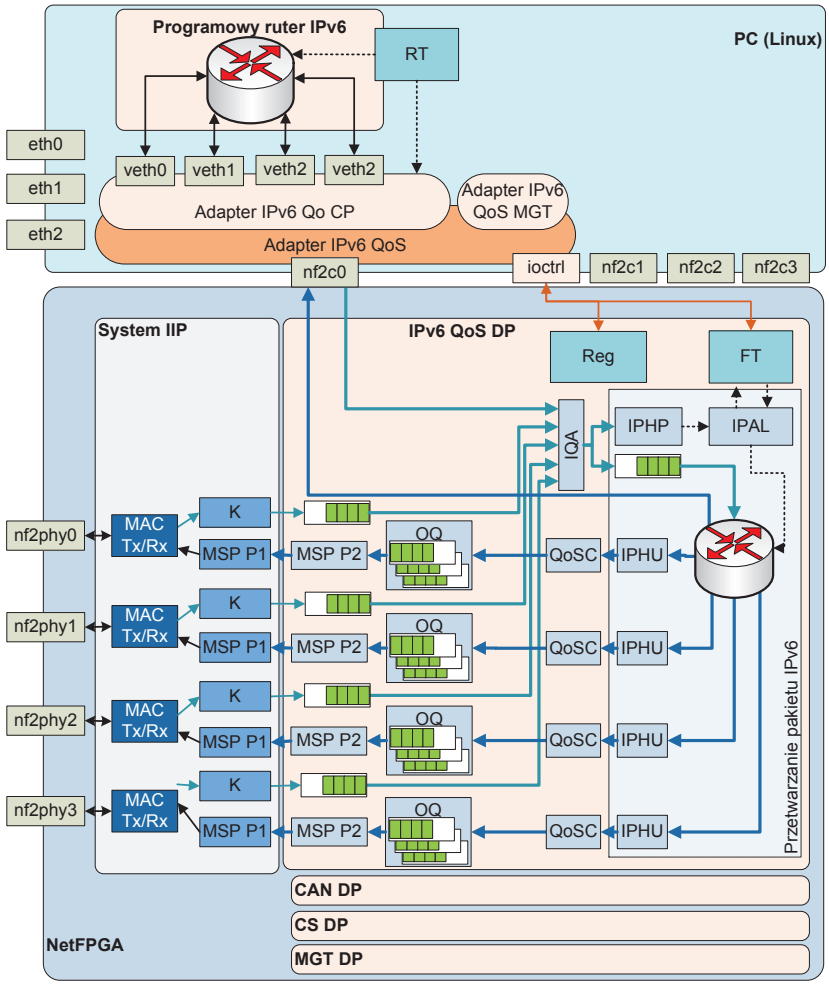
Na Rys. 6 przedstawiono koncepcję implementacji węzła IPv6 QoS z wykorzystaniem platformy NetFPGA. W układzie FPGA jest zrealizowany sprzętowo fragment ścieżki danych węzła IPv6 QoS (IPv6 QoS DP). Blok ten jest połączony z podstawowymi modułami węzła IIP poziomu L1/2 także zaimplementowanymi w układzie FPGA: modułem klasyfikatora (K) i modułem mechanizmu szeregowania pakietów (MSP P1).

Moduł K klasyfikuje pakiety IIP odbierane na czterech portach fizycznych nf2phy0 : nf2phy3 i przekazuje pakiety RI IPv6 QoS do odpowiednich buforów bloku IPv6 QoS DP. Pakiety pobierane są z tych buforów przez moduł arbitra wejściowego (IQA) zgodnie z algorytmem szeregowania round robin i przekazywane do sekcji przetwarzania pakietów. W bloku analizy nagłówka IP (IPHP) następuje wyodrębnienie informacji o adresie docelowym i czasie życia pakietu. Dane te są wykorzystane przez moduł IPAL do podjęcia decyzji o dalszej trasie pakietu. Moduł ten wykorzystuje tablicę trasowania FT. W przypadku, gdy w tablicy FT nie istnieje wpis pozwalający wyznaczyć trasę pakietu, lub gdy czas życia pakietu się skończył pakiet jest przekazywany przez wirtualny interfejs nf2c0 do rutera programowego. Pakiety, dla których można wyznaczyć trasę na podstawie tablicy FT przekazywane są na odpowiedni interfejs fizyczny. Nagłówki takich pakietów są uaktualniane przez moduł IPHU. Następnie są one klasyfikowane przez moduł QoSC na podstawie klas usług i kierowane do odpowiednich kolejek wyjściowych OQ. Moduł szeregowania pakietów MSP P2 realizujący algorytm DRR pobiera pakiety z tych kolejek i w czasie trwania fazy cyklu Systemu IIP przypisanej do RI IPv6 QoS przekazuje do warstwy L1/2 Systemu IIP do modułu MSP P1.

Pakiety przekazywane do rutera programowego są multipleksowane przez programowy moduł adaptera IPv6 QoS i kierowane na wirtualne interfejsy veth0 : veth3 odpowiadające fizycznym interfejsom nf2phy0 : nf2phy3. Są one przetwarzane przez programowy ruter i kierowane na powrót do karty NetFPGA przez wirtualne interfejsy, gdzie są przekazywane na odpowiednie interfejsy fizyczne.

Wybrane wpisy z tablicy rutingu RT są kopiowane do części sprzętowej systemu, do tablicy FT. Tego typu mechanizm sprawia, że część sprzętowa dokonuje przetwarzania większości pakietów w sposób niewidoczny dla rutera programowego, który jest odpowiedzialny tylko za obsługę szczególnych przypadków.

Płaszczyzna sterowania wirtualnego węzła RI IPv6 QoS steruje pracą części sprzętowej poprzez ustalanie parametrów takich modułów jak blok szeregowania pakietów MSP P2, czy kolejki wyjściowe OQ przez mechanizm sprzętowych rejestrów Reg zaimplementowanych na platformie NetFPGA.

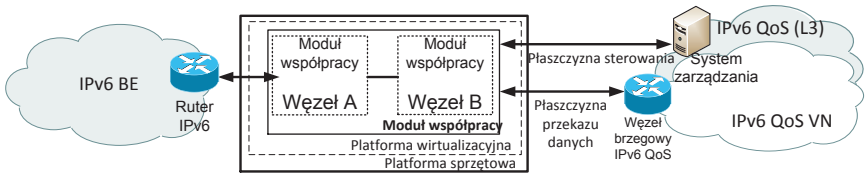



**Rysunek 6.** Koncepcja realizacji węzła IPv6 QoS (wyjaśnienie oznaczeń w tekście)

## 5 Współpraca RI IPv6 QoS z siecią best effort

Aby umożliwić komunikację pomiędzy Równoległym Internetem IPv6 QoS, a zewnętrznym systemem sieciowym IPv6 best effort (IPv6 BE), węzeł EZappliance (pełniący rolę rutera brzegowego tego Równoległego Internetu), został uzupełniony o moduł współpracy.

Moduł ten odpowiada za przekazywanie pomiędzy łączonymi systemami zarówno informacji sterujących (w szczególności informacji o dostępnych w ich obszarze prefiksach IPv6), jak i strumieni danych użytkownika. W tym celu musi on




**Rysunek 7.** Architektura modułu współpracy

funkcjonować jako integralny element obu łączonych systemów, tak w płaszczyźnie sterowania jak i przekazu danych. O ile w przypadku Równoległego Internetu IPv6 QoS mechanizmy funkcjonowania systemu są dokładnie określone, to spotykane obecnie zewnętrzne systemy IPv6 mogą charakteryzować się bardzo zróżnicowanymi wymogami dotyczącymi mechanizmów sieciowych, które muszą być obsługiwane przez należące do nich węzły. Powyższe wymagania zaowocowały opracowaniem architektury modułu współpracy, która została przedstawiona na Rys. 7.

Moduł współpracy składa się z 2 węzłów logicznych, z których jeden jest zintegrowany w pełni z zewnętrznym systemem IPv6 BE, a drugi z systemem RI IPv6 QoS. Oba węzły pracują na platformie wirtualizacji, co umożliwia łatwą i efektywną implementację architektury. Rozwiązanie pozwala na uzyskanie maksymalnej autonomii działania mechanizmów obu łączonych systemów, koniecznych do implementacji w obrębie modułu współpracy.

Komunikacja pomiędzy łączonymi systemami realizowana przez moduł w płaszczyźnie sterowania, sprowadza się do przekazania odpowiedniej informacji rutingowej, a konkretnie informacji o prefiksach IPv6 osiągalnych w obszarze każdego z łączonych systemów. Z kolei w płaszczyźnie przekazu danych istnieje konieczność zapewnienia zgodności informacji przekazywanej pomiędzy łączonymi systemami, co wymaga odpowiedniego dostosowania formatu pakietów przesyłanych w obu łączonych systemach.

Węzeł logiczny A odpowiada za integrację z siecią IPv6 BE i może podlegać pewnym modyfikacjom w celu zapewnienia obsługi maksymalnie szerokiej gamy sieci zewnętrznych. W szczególności obsługuje konieczne interfejsy warstwy łącza danych, protokoły rutingu, itp.

Węzeł logiczny B jest ściśle zintegrowany z systemem IPv6 QoS, a w szczególności z węzłem brzegowym sieci (EZappliance) w płaszczyźnie przekazu danych oraz sterowania. Wykorzystuje on dedykowane interfejsy Ethernet w celu przenoszenia ruchu każdej z powyższych płaszczyzn. Integracja z płaszczyzną sterowania IPv6 QoS jest realizowana z wykorzystaniem interfejsu typu web-service, pozwalającego systemowi zarządzania na kontrolę nad modułem współpracy. Interfejs ten odpowiada również za dwukierunkową wymianę informacji rutingowej pomiędzy modułem, a systemem IPv6 QoS.

Zadanie dostosowania formatu pakietów IPv6 wymienianych między systemami realizowane jest również w węźle B, z użyciem mechanizmów systemu Linux: Netfilter / Ip6tables / NFQueue, pozwalających na daleko posuniętą ich modyfikację.

Moduł współpracy został zaimplementowany w dwóch wersjach. Pierwsza z nich oparta jest na systemie operacyjnym Linux (dystrybucja Fedora 16, jądro 3.1.0), który używany jest w obu węzłach logicznych (A i B). Jest to wersja dostosowana do wdrożenia na popularnych platformach wirtualizacyjnych jak np. Xen, VMWare czy Hyper-V.

Druga wersja stanowi zintegrowane rozwiązanie dedykowane popularnej platformie sprzętowej Mikrotik RouterBoard (model 433AH) i wykorzystuje standardowy dla niej system operacyjny RouterOS (w wersji 5.13) do realizacji węzła A, oraz dystrybucję systemu Linux OpenWRT Backfire (w wersji 10.03.01) uruchamianą w postaci wirtualnej maszyny obsługiwanej przez mechanizm Metarouter systemu RouterOS, do realizacji węzła B.

## 6 Podsumowanie

Artykuł przedstawia opis zagadnień związanych z implementacją węzłów sieci IPv6 QoS na urządzeniach umożliwiających wirtualizację. Jako urządzenia dla węzłów sieci szkieletowej architektury DiffServ wybrano dwie platformy NetFPGA i XEN, zaś dla węzłów brzegowych platformę EZappliance. Ostatnia z platform cechuje się wieloma mechanizmami QoS wbudowanymi w podstawowe moduły platformy. Również na platformie XEN zastosowano wbudowane mechanizmy dla różnicowania przekazu pakietów w sieci, jak DRR oraz klasyfikatory. W pełni oryginalną implementację wszystkich mechanizmów QoS zastosowano dla węzła implementowanego na platformie NetFPGA.

## Literatura

1. ITU-T Rec. Y.3001, „Future Networks: Objectives and design goals", ITU Telecommunication Standardisation Sector, May 2011.
2. W. Burakowski, H. Tarasiuk, A. Beben, G. Danilewicz, „Virtualized network infrastructcture supporting co-existence of Parallel Internet", In Proc. of SNPD 2012, Kyoto, August 2012, pp. 679-684.
3. H. Tarasiuk, et al., „Architektura i mechanizmy równoległego Internetu IPv6 QoS", Przegląd Telekomunikacyjny, SIGMA NOT, 2011, nr 8-9, t. CD-ROM, s.944-954.
4. S. Blake i in., „An Architecture for Differentiated Services", Internet RFC 2475, 10/1998.
5. ITU-T Rec. Y.2001, „ITU-T, General overview of NGN", ITU Telecommunication Standardisation Sector, December 2004.
6. P. Papadimitriou, et al., „Implementing network virtualization for a future internet", in Proc. of 20th ITC Specialist Seminar, Hoi An, Vietnam, May 2009.
7. N. Mckeown, et al., „Openflow: Enabling innovation in campus networks", white paper, 2008.
8. EZappliance: http://www.ezchip.com.

9. J. W. Lockwood, et al., „NetFPGA - An Open Platform for Gigabit-rate Network Switching and Routing", In Proc. of the 2007 IEEE International Conference on Microelectronic Systems Education, San Diego, June 2007.
10. XEN project: http://www.xen.org.
11. H. Tarasiuk, et al., „A Proposal of the IPv6 QoS System Implentation in Virtual Infrastructure", In Prof. of Networks 2012, Rzym 2012.
12. H. Tarasiuk, J. Rogowski, „On the Signaling System in the IPv6 QoS Parallel Internet", In Proc. of CSNDSP 2012, Poznan 2012.
13. Quagga project: http://www.quagga.net/.
14. Linux Advanced Routing & Traffic Control, http://lartc.org/
15. M. Shreedhar, G. Varghese, Efficient Fair Queuing Using Deficit Round-Robin, IEEE/ACM Transactions on Networking, Vol. 4 Issue 3, 1996.
16. A. Chydzinski, et al., „Virtualization Devices for Prototyping of Future Internet", In Proc. of SNPD 2012, Kyoto, August 2012, pp. 672-678.

# Aplikacje testowe i ich odwzorowanie na klasy usług w Równoległym Internecie IPv6 QoS

Halina Tarasiuk[1], Wojciech Burakowski[1], Marek Natkaniec[2], Janusz Gozdecki[2], Krzysztof Łoziak[2], Katarzyna Kosek-Szott[2], Szymon Szott[2], Adam Grzech[3], Paweł Świątek[3], Bartosz Lewandowski[4], Cezary Mazurek[4], Arkadiusz Radziuk[4], Dariusz Walczak[4]

[1] Politechnika Warszawska
[2] AGH Akademia Górniczo-Hutnicza w Krakowie
[3] Politechnika Wrocławska
[4] Poznańskie Centrum Superkomputerowo-Sieciowe

**Streszczenie** W artykule przedstawiono propozycję odwzorowania wymagań aplikacji testowych na klasy usług koniec-koniec w Równoległym Internecie (RI) IPv6 QoS. Klasy usług koniec-koniec są klasami sieciowymi, których definicje obejmują m.in. wymagania QoS na przekaz pakietów przez sieć, mechanizmy QoS w poszczególnych węzłach sieci, mechanizmy przyjmowania nowych wywołań. Aplikacje testowe dla RI IPv6 QoS, to głównie aplikacje Internetu Rzeczy, w tym aplikacje e-zdrowie, zarządzania energią, bezpieczeństwa publicznego oraz zdalnego nauczania.

## 1 Wprowadzenie

W artykule przedstawiono propozycję odwzorowania aplikacji testowych na klasy usług w Równoległym Internecie (RI) IPv6 QoS [1]. Architektura RI IPv6 QoS [2] bazuje na architekturze NGN (Next Generation Networks) [3] oraz DiffServ (Differentiated Services) [4] i umożliwia dzięki mechanizmom przyjmowania nowych wywołań oraz mechanizmom QoS w sieci IP ścisłe gwarancje jakości przekazu pakietów przez sieć w ramach wybranych połączeń aplikacji. Aplikacje testowe posiadają mechanizmy systemu sygnalizacji umożliwiające przesyłanie żądań QoS do sieci [5]. W celu obsługi żądań przez sieć zdefiniowano zestaw klas usług koniec-koniec oraz odwzorowania tych klas na przykładowe aplikacje testowe, a także klasy usług wspierane przez poszczególne sieci dostępowe oraz sieć szkieletową IP. Każde żądanie jest przypisane do odpowiedniej klasy usług typu koniec-koniec. Klasa usług koniec-koniec jest definiowana poprzez wymagania na jakość przekazu pakietów przez sieć od końca do końca dla grupy obsługiwanych połączeń. Połączenia przyjęte do obsługi nie powinny zmieniać jakości przekazu pakietów w ramach danej klasy poniżej wartości granicznych wymagań QoS dla tejże klasy. Aby zrealizować zdefiniowane wymagania QoS, klasy usług koniec-koniec są odpowiednio odwzorowane do klas usług obsługiwanych przez poszczególne typy sieci oraz ich węzły.

Aplikacje testowe, wybrane dla RI IPv6 QoS, to aplikacje z grupy tzw. aplikacji Internetu Rzeczy (Internet of Things, IoT), są to aplikacje e-zdrowie, zarządzanie energią, bezpieczeństwo publiczne, zdalne nauczanie (w tym wideokonferencja wysokiej rozdzielczości – High Definition). Mechanizmy gwarancji jakości przekazu pakietów przez sieć są mechanizmami zdefiniowanymi w ramach architektury DiffServ [4]. W odróżnieniu od dotychczas proponowanych klas usług koniec-koniec dla aplikacji multimedialnych typu VoIP, czy IPTV [6], aplikacje z grupy Internetu Rzeczy, które generują dane np. z czujników (sensorów) wymagają definicji klasy usług koniec-koniec z wymaganiami na małe opóźnienia przekazu pakietów w sieci dla połączeń obsługiwanych przez protokół transportowy TCP. Dlatego też dla RI IPv6 QoS zostały zdefiniowane mechanizmy QoS dla klasy usług koniec-koniec typu Low Latency Data [7].

Artykuł jest zorganizowany następująco. W rozdziale 2. przedstawiamy definicje klas usług typu koniec-koniec, klas usług dla sieci IP, a także klas usług w sieci dostępowej WiFi oraz odwzorowanie klas usług na przykładowe aplikacje. Rozdział 3. zawiera odwzorowania scenariuszy aplikacji testowych dla RI IPv6 QoS na klasy usług koniec-koniec. Rozdział 4. stanowi podsumowanie.

## 2 Klasy usług

Dla RI IPv6 QoS zdefiniowano klasy usług bazując na propozycji [6] oraz zaleceniach IETF [7] i [8]. W rozdziale przedstawiono definicje klas usług koniec-koniec. Następnie odwzorowanie tych klas na klasy usług w sieci IP oraz w sieci dostępowej WiFi.

### 2.1 Klasy usług koniec-koniec

Tab. 1 przedstawia odwzorowanie przykładowych typów aplikacji w ramach RI IPv6 QoS na klasy usług koniec-koniec oraz wymagania QoS w relacji koniec-koniec. Ponadto, dla poszczególnych klas usług koniec-koniec zaproponowano wartości pola IPv6 Traffic Class dla klasyfikacji pakietów IPv6 do poszczególnych klas usług w sieci. RI IPv6 QoS wspiera następujące klasy usług koniec-koniec:

- Network Control, Signalling – odpowiadają za obsługę odpowiednio pakietów sterowania siecią i pakietów systemu sygnalizacji;
- Telephony, Real Time (RT) Interactive – klasy realizują obsługę pakietów w ramach połączeń aplikacji wymagających reżimu czasu rzeczywistego;
- MM Streaming – obsługuje aplikacje z protokołem UDP bez reżimu czasu rzeczywistego;
- High Throughput Data, Low Latency Data – obsługuje aplikacje z protokołem TCP bez reżimu czasu rzeczywistego;
- Sieć obsługuje również usługę Standard typu best effort bez gwarancji QoS.

**Tablica 1.** Odwzorowanie przykładowych aplikacji na klasy usług koniec-koniec oraz wymagania QoS w sieci IPv6 QoS

| **Przykładowe aplikacje** | **Klasy usług koniec-koniec** | **Pole IPv6 Traffic Class** | **Wymagania QoS koniec-koniec na poziomie sieci** | | |
|---|---|---|---|---|---|
| | | | **IPLR** | **średnie IPTD** | **IPDV** |
| Sterowanie siecią | Network Control | 101000 | $10^{-3}$ | 100 ms | — |
| Sygnalizacja | Signalling | | | | |
| Połączenia głosowe | Telephony | 101110 | $10^{-3}$ | 100/350 ms (lokalne/długie odległości) | 50 ms |
| Wideokonferencja | RT Interactive | 100000 | $10^{-3}$ | 100/35 0ms (lokalne/długie odległości) | 50 ms |
| Wideo na żądanie | MM Streaming | 11010<br>11100<br>11110 | $10^{-3}$ | 1 s<br>nie jest krytyczne | — |
| FTP | High Throughput Data | 1010<br>1100<br>1110 | $10^{-3}$ | 1 s<br>nie jest krytyczne | — |
| Dane sensorowe | Low Latency Data | 110000 | $10^{-3}$ | 400 ms | |
| Best effort | Standard | 0 | — | — | — |

### 2.2 Klasy usług w sieci IP

W sieci IP wspierane są klasy usług zarówno w ruterach brzegowych sieci, jak i w ruterach szkieletowych. Tab. 2 przedstawia odpowiednio propozycję odwzorowania przykładowych aplikacji na klasy usług koniec-koniec, klasy usług w ruterach brzegowych oraz w ruterach szkieletowych. W odróżnieniu od ruterów brzegowych (zaimplementowanych na urządzeniach EZappliance [9]), w ruterach szkieletowych (węzły sieci zaimplementowane na urządzeniach NetFPGA i XEN [9]) zaproponowano agregację klas usług dla aplikacji wymagających obsługi połączeń z reżimem czasu rzeczywistego do jednej klasy usług Real Time (RT). Ponadto, dla węzłów bazujących na technologii NetFPGA dokonano agregacji klas usług dla aplikacji nie wymagających czasu rzeczywistego do wspólnej klasy Non real time (NRT), ze względu na ograniczenia wynikające z implementacji systemu kolejkowego. Brak agregacji usług w ruterach brzegowych jest uwarunkowany ograniczeniami na obsługę określonego formatu danych w ruterach brzegowych implementowanych na urządzeniu EZappliance. Dla aplikacji z grupy wymagających reżimu czasu rzeczywistego w ruterach brzegowych zastosowano mechanizmy monitorowania ruchu typu policer, zaś dla aplikacji bez reżimu czasu rzeczywistego mechanizm kształtowania ruchu. Szczegóły związane z zastosowanymi mechanizmami QoS na poziomie warstwy pakietowej przedstawiono w [9]. Szczegóły związane z wymiarowaniem klas usług w sieci IP zostały przedstawione w [2]. W tym przypadku sieć szkieletowa IP jest przewymiarowana, zaś dla zasobów w ruterach brzegowych jest zastosowany mechanizm przyjmowania nowych wywołań obsługiwany przez system sygnalizacji [5]. Algorytmy przyjmowania nowych wywołań dla RI IPv6 QoS bazują na propozycji [6] dla sieci IP. Ponadto, w ramach poszczególnych klas usług koniec-koniec zdefiniowano dwa typy połączeń, czyli połączenia planowane i na żądanie [5]. Należy podkreślić, że wiadomości sygnalizacyjne oraz sterowania siecią są obsługiwane przez klasę usług koniec-koniec Signalling, która nie wymaga stosowania algorytmów przyjmowania nowych wywołań.

### 2.3 Klasy usług w sieci WiFi

W bezprzewodowej sieci dostępowej, tzn. pomiędzy urządzeniem typu punkt dostępu AP (Access Point) oraz terminalem klienta odpowiednia jakość obsługi będzie realizowana z użyciem funkcji EDCA (Enhanced Distributed Channel Access) standardu IEEE 802.11. W celu różnicowania obsługi, funkcja EDCA definiuje cztery kategorie dostępu AC (Access Categories). W przeciwieństwie jednak do standardu IEEE 802.11, proponowana przez nas funkcja EDCA wspiera tzw. ścisłe priorytety dla obsługi strumieni głosowych oraz wideo. Oznacza to, że ramki należące do kategorii głosowej mogą rywalizować jedynie z innymi ramki kategorii głosowej Vo (Voice). Identyczna sytuacja występuje dla ramek ruchu wideo Vi (Video), BE (Best Effort) oraz ruchu tła BK (Background). Funkcjonalność ta została zrealizowana poprzez odpowiednią konfigurację parametrów funkcji dostępu do kanału radiowego w standardzie IEEE 802.11. Odpowiednie odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz kategoriami dostępu w standardzie IEEE 802.11 zostało przedstawione w Tab. 3.

**Tablica 2.** Odwzorowanie przykładowych aplikacji oraz klas usług koniec-koniec na klasy usług w ruterach brzegowych i szkieletowych sieci IP

| **Aplikacja** | **Klasy usług koniec-koniec** | **Klasy usług w ruterach brzegowych (EZappliance)** | **Klasy usług w ruterach szkieletowych (XEN)** | **Klasy usług w ruterach szkieletowych (NetFPGA)** |
|---|---|---|---|---|
| Voice over IP | Telephony | Telephony | Real time (RT) | Real time (RT) |
| Video conferencing | RT Interactive | RT Interactive | | |
| Signalling and management messages | Signalling | Signalling | Signalling | Signalling |
| Video on demand | MM Streaming | MM Streaming | MM Streaming | Non real time (NRT) |
| FTP | High Throughput Data | HTD | HTD | |
| Sensor monitoring | Low Latency Data | LLD | LLD | |
| Best effort | Standard | Standard | STD | STD |

**Tablica 3.** Odwzorowanie pomiędzy klasami usług w sieci IPv6 QoS oraz kategoriami dostępu w standardzie IEEE 802.11

| **EDCA AC** | **Klasa usług koniec-koniec** | **AIFSN** | **CWMIN** | **CWMAX** | **TXOP$_{\mathbf{Limit}}$** $[\mu s]$ |
|---|---|---|---|---|---|
| Vo | Telephony<br>Real time Interactive | 2 | 7 | 15 | 3264 |
| Vi | MM Streaming<br>High Throughput Data<br>Low Latency Data | 17 | 15 | 31 | 6016 |
| BE | Signalling | 48 | 31 | 1024 | 0 |
| BK | STD | 52 | 31 | 1024 | 0 |

### 2.4 Odwzorowanie klas w sieci IP i WiFi na klasy usług koniec-koniec

Ponieważ funkcja EDCA zawiera jedynie cztery kolejki sprzętowe, przez co różnicuje jedynie cztery klasy ruchu (zdefiniowane w rozdziale 2.3), konieczne stało się wprowadzenie odwzorowania pomiędzy klasami usług zdefiniowanymi w sieci IPv6 QoS w relacji koniec-koniec oraz kategoriami dostępu funkcji EDCA standardu IEEE 802.11. Pierwszą kategorię usług charakteryzuje bardzo duża wrażliwość na opóźnienie oraz jego zmienność. Są to usługi czasu rzeczywistego takie jak np. VoIP oraz wideokonferencja, które należą do klasy usług (w relacji koniec-koniec): telefonia oraz usługi interaktywne czasu rzeczywistego. Usługi te zostały więc przyporządkowane do kolejki sprzętowej funkcji EDCA o najwyższym priorytecie, tzn. AC_Vo. Do drugiej kolejki sprzętowej w punkcie dostępu AP, tzn. AC_Vi, należą takie klasy usług jak: strumieniowanie multimediów, transfer danych o dużej przepływności oraz transfer danych o małym opóźnieniu transmisji. Przykłady aplikacji dla usług tej klasy to: wideo na żądanie, FTP, oraz dane pochodzące z różnego rodzaju czujników. Kolejna kolejka sprzętowa funkcji EDCA, tzn. AC_BE, obejmuje takie klasy jak sygnalizacja oraz ruch związany z transmisją danych zarządzających przesyłany w celu zapewnienia odpowiedniego sterowania w sieci. Ostatnia kolejka sprzętowa, a więc AC_BK zawiera klasę usług w relacji koniec- koniec standard, której przykładem są wszystkie aplikacje korzystające dotychczas z klasy best-effort. Tab. 4 podsumowuje dotychczasowe założenia i przedstawia przyjęte odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz kategoriami dostępu funkcji EDCA standardu IEEE 802.11.

**Tablica 4.** Odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz kategoriami dostępu funkcji EDCA standardu IEEE 802.11

| **Przykładowe aplikacje** | **Klasy usług w relacji koniec-koniec** | **Klasy usług w bezprzewodowej sieci dostępowej** |
|---|---|---|
| Połączenia głosowe | Telephony | Real Time (AC_Vo) |
| Wideokonferencja | RT Interactive | |
| Sygnalizacja | Signalling | Signalling (AC_BE) |
| Sterowanie siecią | Network Control | |
| Wideo na żądanie | MM Streaming | Non Real Time (AC_Vi) |
| FTP | High Throughput Data | |
| Dane sensorowe | Low Latency Data | |
| Best effort | Standard | Standard (AC_BK) |

# 3 Wymagania aplikacji i odwzorowanie na klasy usług koniec-koniec

Odwzorowanie aplikacji na odpowiednie klasy usług koniec-koniec zależy od wymagań na jakość przekazu w ramach poszczególnych połączeń aplikacji.

## 3.1 Aplikacje e-zdrowie

Zdalne systemy monitoringu e-zdrowie są bardzo ważnym narzędziem w dziedzinie ochrony zdrowia. Głównym celem dla proponowanych systemów e-zdrowie jest zdalne zbieranie sygnałów monitorujących stan zdrowia pacjenta i parametry otoczenia mogące wpływać na jego zdrowie oraz przesłanie ich do zdalnego serwera medycznego. Lekarze specjaliści mogą postawić diagnozę opartą na aktualnych odczytach z czujników oraz na historii choroby pacjenta zapisanej na serwerach medycznych i w sposób optymalny prowadzić pacjenta. W ramach projektu zostały zdefiniowane i zaimplementowane 3 testowe aplikacje e-zdrowie: Asthma, eDiab i SmartFit.

### Asthma

W tej części zostaną przedstawione wymagania aplikacji Asthma oraz ich odwzorowanie na klasy usług w relacji koniec-koniec. Aby zapewnić prawidłowe działanie aplikacji konieczne było zdefiniowanie pięciu różnych scenariuszy takich jak: monitorowanie pracy serca, wykrywanie świstów, pomiar efektywności płuc, monitoring jakości powietrza oraz konsultacje. Dla różnych przypadków choroby wyróżniono dwa rodzaje konsultacji: głosową oraz wideokonferencję (w niskiej oraz wysokiej jakości). Każdy ze scenariuszy posiada inne wymagania transmisyjne, a co za tym idzie różne wymagania QoS. Pierwszym krokiem jest identyfikacja i uwierzytelnianie pacjenta oraz przesłanie informacji o jego lokalizacji na serwer e-zdrowie. Wbudowany odbiornik GPS w urządzeniu mobilnym pacjenta gwarantuje bardzo dokładne szacowanie miejsca pobytu pacjenta. Na podstawie danych lokalizacyjnych pacjenta z serwerów zewnętrznych pobierana jest informacja o jakości powietrza (np. stężenie Cox, SO2, NO, O3, PM10, PM2.5). Informacja o zanieczyszczeniu jest analizowana na serwerze e-zdrowie, jak również przesyłana do pacjenta wraz z informacjami o fakcie przekroczenia dopuszczalnej wartości jakiegokolwiek z mierzonych parametrów. Ponadto, jeśli dla lokalizacji pacjenta nie można uzyskać danych o zanieczyszczeniu powietrza ze zdalnych serwerów, to do pomiaru zanieczyszczeń można użyć dostępnych na rynku mierników przenośnych, które zazwyczaj mają mniejszą precyzję pomiarów. Należy również podkreślić, że informacje na temat zanieczyszczeń nie są konieczne do przeprowadzenia prawidłowej diagnozy stanu zdrowia pacjenta, ale są bardzo ważne ze względu na zalecenia dotyczące aktywności ruchowej pacjenta w miejscu pobytu. Następnie wykonywane są pomiary wydajności płuc (PEF/FEV1), szmerów w płucach i rytmu serca. Pomiary są wysyłane przez Bluetooth do urządzenia mobilnego pacjenta, a następnie, za pośrednictwem jednej

z dostępnych technologii bezprzewodowych, które obsługują Internet typu IPv6 QoS, pomiary są przesyłane na serwer e-zdrowie. Serwer przekazuje wszystkie niezbędne dane do lekarza. Na podstawie tych danych lekarz wystawia diagnozę i zalecenia dotyczące dalszego postępowania. Jeśli tylko lekarz zatwierdzi diagnozę, pacjent natychmiast otrzymuje informację o swoim stanie zdrowia i zalecanych lekach. W razie potrzeby lekarz może również zainicjować poprzez serwer e-zdrowie zdalną konsultację, która wykorzystuje aplikację audio lub wideo konferencji. W tabeli 5 przedstawiono odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz przyjętymi scenariuszami dla aplikacji Asthma. Należy zauważyć, iż wszystkie połączenia realizowane dla aplikacji Asthma są połączeniami planowanymi.

**Tablica 5.** Odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz scenariuszami dla aplikacji Asthma

| **Scenariusz aplikacyjny** | **Klasy usług w relacji koniec-koniec** | **Wymagania QoS na poziomie aplikacji** |
|---|---|---|
| Monitorowanie jakości powietrza | Standard | Brak wymagań |
| GPS | Standard | Brak wymagań |
| Monitorowanie pracy serca | Low Latency Data | Opóźnienie: małe, Jitter: nieistotny, Straty: 0% Średnia szybkość bitowa: - „w dół": 5 kb/s, - „w górę": 50 kb/s |
| Wykrywanie świstów | Low Latency Data | Opóźnienie: małe., Jitter: nieistotny, Straty: 0% średnia szybkość bitowa: - „w dół": 5 kb/s, - „w górę": 100 kb/s |
| Pomiar efektywności płuc | Low Latency Data | Opóźnienie: małe, Jitter: nieistotny, Straty: 0% średnia szybkość bitowa: - „w dół": 5 kb/s, - „w górę": 20 kb/s |
| Konsultacje głosowe | Telephony | Opóźnienie: < 100ms, Jitter: < 50 ms, Straty: < 2%, Maksymalna szybkość bitowa: - „w dół": 80kb/s, - „w górę": 80kb/s |
| Konsultacje wideo o niskiej jakości | Real Time Interactive | Opóźnienie: < 400ms, Jitter: < 50 ms, Straty: < 2%, Maksymalna szybkość bitowa: - „w dół": 128kb/s, - „w górę": 128kb/s |
| Konsultacje wideo o wysokiej jakości | Real Time Interactive | Opóźnienie: ¡ 400ms, Jitter: < 50 ms, Straty: < 1%, Maksymalna szybkość bitowa: - „w dół": 1Mb/s, - „w górę": 1Mb/s |

### eDiab

Celem aplikacji eDiab jest wspomaganie terapii cukrzycy typu 2 poprzez realizację zadań: przesyłania, przetwarzania i archiwizacji danych związanych ze stanem pacjenta. Realizacja zadań aplikacji wiąże się z przesyłaniem określonych

rodzajów danych. Najważniejszymi danymi pomiarowymi są stężenie glukozy we krwi i ciśnienie. Pomiary wykonywane są kilka razy dziennie, mają one niewielki rozmiar (ok. 1 kb). W skrajnym przypadku użytkownik będzie wykonywać nie więcej niż jeden pomiar na pięć minut. Do realizacji scenariuszy uwzględniających aktywność fizyczną potrzebna jest obsługa danych pomiarowych pulsu. W tym przypadku dane mogą być zbierane i wysyłane nawet w odstępach kilku sekund. Rozmiar pakietu nadal jest niewielki (ok. 1kb). W ramach obsługi sytuacji krytycznej mierzone jest stężenie glukozy, sygnał GPS, nawiązywane jest połączenie głosowe i przesyłane są zarchiwizowane dane pacjenta. Dodatkowo, na potrzeby sygnalizacji, konfiguracji połączeń i negocjacji ich parametrów, wysyłane są pojedyncze wiadomości składające się z pojedynczego pakietu. W tabeli 6 przedstawiono odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz scenariuszami dla aplikacji eDiab. Należy zauważyć, iż wszystkie połączenia realizowane dla aplikacji eDiab są połączeniami planowanymi.

**Tablica 6.** Odwzorowanie pomiędzy klasami usług koniec-koniec w sieci IPv6 QoS oraz scenariuszami dla aplikacji eDiab

| **Scenariusz aplikacyjny** | **Klasy usług w relacji koniec-koniec** | **Protokół transportowy** | **Wymagana szybkość bitowa na poziomie aplikacji** |
|---|---|---|---|
| Pomiar stężenia glukozy | Low Latency Data | TCP | Średnia szybkość bitowa: 1kb/s |
| Pomiar ciśnienia | Low Latency Data | TCP | Średnia szybkość bitowa: 1kb/s |
| Pomiar pulsu | Low Latency Data | TCP | Średnia szybkość bitowa: 1kb/s |
| GPS | Low Latency Data | TCP | Średnia szybkość bitowa: 1kb/s |
| Połączenie głosowe | Telephony | UDP | Maksymalna szybkość bitowa: 64kb/s |
| Dane archiwalne | High Throughput Data | TCP | Średnia szybkość bitowa: 100kb/s |
| Komunikaty aplikacji | Signalling | TCP | Wymagania zostaną opracowane na podstawie badań |

### SmartFit

SmartFit to narzędzie zaprojektowane w celu wspomagania planowania i monitorowania treningu wytrzymałościowego i technicznego sportowca. Jest to aplikacja przeznaczona zarówno dla sportowców jak i trenerów. Dostępne w aplikacji funkcjonalności związane są ze scenariuszami, które składają się z usług. Oprócz usług związanych z przetwarzaniem danych, dostępny jest również zestaw usług związanych z transferem danych. Transfer danych w aplikacji SmartFit jest bezprzewodowy. W zależności od scenariusza przesyłane są różne dane pomiarowe

i różne mogą być również wymagania dotyczące ich transferu. Podstawowy scenariusz przygotowany dla aplikacji SmartFit związany jest z przesyłaniem bieżących pomiarów pulsu sportowca podczas treningu. Dane te mogą być przydatne zarówno do oceny zaangażowania sportowca podczas treningu jak i do oceny czy cele treningowe zostały zrealizowane. Cechą charakterystyczną tego scenariusza jest to, że stawia niewielkie wymagania dotyczące szybkości bitowej dla przesłania danych (tabela 7). Bardziej wymagający pod tym względem jest drugi z opracowanych scenariuszy, który dotyczy wspomagania treningu technicznego. W tym scenariuszu przesyłane oraz przetwarzane są dane z czujników rozmieszczonych na ciele użytkownika: przyspieszenia oraz EMG. W zależności od konfiguracji czujników pomiarowych, średnia szybkość bitowa przesyłanych danych z jednego czujnika może dochodzić do 64 kb/s. Dodatkowo, w ramach rozpatrywanego scenariusza, możliwe jest przesyłanie danych audio i wideo, które pozwalają na komunikację pomiędzy sportowcem a jego trenerem podczas treningu (tabela 7). Ostatni z opracowanych scenariuszy umożliwia przesyłanie pomiarów pulsu, częstości oddechów oraz temperatury ciała. Wymagania odnośnie szybkości bitowych przesyłania danych nie są duże. To co odróżnia ten scenariusz od wcześniej omówionych to jego przeznaczenie: sytuacja krytyczna. W takich warunkach najważniejsze jest to, by dane zostały przesłane z wymaganą szybkością bitową. Jest to najbardziej wymagający z opracowanych scenariuszy (tabela 7). W sieci, ruch jest generowany również przez komunikaty przesyłane przez aplikację: system sygnalizacji. Są to niewielkie komunikaty, które służą do konfiguracji połączeń i negocjacji ich parametrów. Wszystkie połączenia realizowane dla aplikacji SmartFit są połączeniami planowanymi.

### 3.2 Aplikacja zarządzanie energią

Aplikacja HomeNetEnergy [10] realizuje scenariusze z obszaru Internetu Rzeczy [11] związane z monitoringiem oraz inteligentnym zarządzaniem energią elektryczną w rozproszonych lokalizacjach z wykorzystaniem informacji kontekstowej oraz profili użytkowników. Aplikacja łączy w sobie zalety tego podejścia oraz protokołu IPv6 z architekturą zorientowaną na zdarzenia - w celu umożliwienia analizy zużycia energii przez urządzenia elektryczne w czasie bliskim do rzeczywistego, ręcznego zarządzania odbiornikami energii przez użytkowników, oraz inteligentnego zarządzania odbiornikami przez samą aplikację. Zaproponowane scenariusze aplikacyjne w ogólności dotyczą dwóch obszarów powszechnie znanych pod pojęciem komunikacja typu maszyna-maszyna (Machine-2-Machine, M2M) [12,13], tj. akwizycji danych z rozproszonych czujników, oraz interakcji z urządzeniami wykonawczymi. Wymagania jakie aplikacja HomeNetEnergy stawia sieci IPv6 QoS, wynikają wprost z koncepcji Internetu Rzeczy oraz M2M. Koncepcja Internetu Rzeczy zakłada, że każdy obiekt w otoczeniu człowieka będzie częścią sieci, posiadając unikalny adres i zdolność do komunikacji z otoczeniem. Biorąc pod uwagę setki milionów urządzeń obecnych w otoczeniu człowieka, potrzebne są mechanizmy mogące je jednoznacznie zaadresować i zintegrować z istniejącą infrastrukturą. Z kolei koncepcja M2M opisuje systemy zdolne do komunikacji

**Tablica 7.** Klasy usług koniec-koniec w aplikacji SmartFit

| **Rodzaj danych** | **Klasy usług** | **Protokół transportowy** | **Wymagana szybkość bitowa na poziomie aplikacji** |
|---|---|---|---|
| Puls | Low Latency Data | TCP | Średnia szybkość bitowa: 10kb/s |
| EMG | Low Latency Data | TCP | Średnia szybkość bitowa: 64kb/s |
| Przyspieszenie | Low Latency Data | TCP | Średnia szybkość bitowa: 64kb/s |
| Częstość oddechów | Low Latency Data | TCP | Średnia szybkość bitowa: 10kb/s |
| Temperatura ciała | Low Latency Data | TCP | Średnia szybkość bitowa: 1kb/s |
| Głos | Telephony | UDP | Maksymalna szybkość bitowa: 80kb/s |
| Wideo | RT Interactive | UDP | Maksymalna szybkość bitowa: 200kb/s |
| Dane archiwalne | High Throughput Data | TCP | Średnia szybkość bitowa: 100kb/s |
| Komunikaty aplikacji | Signalling | TCP | Wymagania zostaną opracowane na podstawie badań |

i działania bez udziału lub z minimalnym udziałem człowieka w obszarze monitoringu i sterowania. Tak działające systemy wymagają, by komunikacja między nimi była wysoce niezawodna i zapewniała właściwe parametry jakości przekazu danych przez sieć, takie jak opóźnienie czy szybkość bitowa. Istotnym wyróżnikiem komunikacji M2M jest charakter ruchu sieciowego, który składa się z bardzo dużej liczby relatywnie małych komunikatów. Jednak biorąc pod uwagę liczbę urządzeń, ruch zagregowany może wymagać dużych szybkości bitowych na poziomie sieci. Ruch generowany przez elementy aplikacji HomeNetEnergy w ogólności można podzielić na dwa rodzaje. Pierwszy z nich jest związany z ręcznym monitorowaniem zużycia energii przez użytkownika. Dla tego scenariusza parametry jakości nie są szczególnie istotne, ponieważ różnica rzędu setek milisekund z punktu widzenia użytkownika jest pomijalnie mała. Drugi rodzaj ruchu jest związany z inteligentnym, maszynowym przetwarzaniem zdarzeń, dla którego niezbędne są gwarancje przekazu danych przez sieć z określoną średnią szybkością bitową. Ruch periodyczny w aplikacji jest generowany w stałych odstępach czasu, podczas gdy ruch zdarzeniowy jest generowany przez czujniki i moduły aplikacji w chwilach wykrycia pewnych zdefiniowanych sytuacji w otoczeniu, lub w wyniku reakcji aplikacji na wystąpienie tych zdarzeń, np. w postaci wysłania komendy do urządzenia wykonawczego. Przykładem ruchu periodycznego może być ruch generowany przez zaawansowany miernik energii elektrycznej. W przeciwieństwie do ruchu periodycznego, nie można precyzyjnie określić momentu wystąpienia zdarzeń, a zatem przewidzieć, kiedy niezbędna będzie transmisja

danych. Jednocześnie tego typu zdarzenia są niezwykle istotne dla prawidłowego działania systemów i muszą być dostarczane z zachowaniem gwarancji szybkości bitowej w sieci. Analizując wymagania aplikacji, jako sieć transmisyjną wybrano Równoległy Internet IPv6 QoS, który dzięki zastosowaniu protokołu IPv6 zapewnia wymaganą pulę adresów dla wszystkich potencjalnych urządzeń, zaś dzięki zastosowanym mechanizmom QoS i klasom usług zapewnia również gwarancję jakości przekazu danych przez sieć. W kontekście mechanizmów komunikacyjnych zapewnianych przez RI IPv6 QoS, aplikacja HomeNetEnergy wykorzystuje dwie klasy usług koniec-koniec. Dla ruchu generowanego w odpowiedzi na żądania użytkownika (interakcja z użytkownikiem) zastosowano klasę Standard. Dla scenariuszy maszynowego przetwarzania zdarzeń i transmisji komend sterujących aplikacja korzysta z dedykowanej dla tego typu zastosowań klasy Low Latency Data. Klasa ta gwarantuje, że dane będą dostarczone ze średnim opóźnieniem nie większym niż 400 ms. Ponadto, we wszystkich scenariuszach aplikacja wykorzystuje protokół TCP, gwarantujący dostarczenie wszystkich pakietów do celu. Połączenia realizowane w ramach aplikacji HomeNetEnergy są połączeniami planowanymi.

### 3.3 Aplikacja bezpieczeństwo publiczne

MobiWatch jest platformą integracyjną dla rozproszonych systemów bezpieczeństwa i monitoringu [14]. Platforma ta integruje dane pochodzące z systemów monitoringu wizyjnego, zarówno z systemów telewizji przemysłowej jak i z nowoczesnych cyfrowych systemów monitoringu, z danymi pochodzącymi z sieci sensorowych. Multimedialne strumienie danych są przesyłane i przetwarzane przez system MobiWatch, a następnie wzbogacane o dodatkowe informacje dotyczące wybranych obiektów, ludzi, czy zdarzeń zachodzących w obszarze monitorowania, oraz o odpowiednie metadane. System umożliwia również dołączanie informacji kontekstowych lub pochodzących z zewnętrznych źródeł danych. Dodatkowe informacje mogą być uzyskane z wykorzystaniem usług analizy audio i wideo, dzięki sensorom zintegrowanym z systemem oraz z dodatkowych baz danych. Platforma MobiWatch wykorzystuje techniki automatycznego przetwarzania zdarzeń oraz paradygmat architektury zorientowanej na usługi [14]. Dzięki tym cechom stosunkowo bezproblemowe jest rozszerzanie systemu o zestaw nowych komponentów przetwarzających dane multimedialne, ich opisy oraz o różnego rodzaju aplikacje klienckie. Dzięki przetwarzaniu zdarzeń z wykorzystaniem techniki CEP (Complex Event Processing) możliwe jest monitorowanie na bieżąco stanu systemu oraz wykrywanie sytuacji podejrzanych i krytycznych minimalizując czas reakcji [15]. Jedną z kluczowych cech systemu MobiWatch jest wykorzystanie sieci IPv6 QoS. W scenariuszach związanych z zapewnieniem bezpieczeństwa wykorzystanie mechanizmów QoS jest kluczowe przy przesyłaniu na żywo obrazu w wysokiej rozdzielczości z miejsca zdarzenia. Ważne jest również zapewnienie jakości przy przesyłaniu danych multimedialnych zarejestrowanych wcześniej, szczególnie, że w związku z coraz większymi rozmiarami multimediów oraz powiększającą się liczbą źródeł wideo, będą one coraz częściej składowane w zdalnych repozytoriach. Przeglądanie archiwalnych materiałów

wiąże się z koniecznością dostępu do dużej liczby plików, najlepiej w taki sposób, żeby dla oglądającego sprawiały wrażenie dostępnych lokalnie. Mechanizm zapewnienia jakości usług w przypadku inteligentnych systemów monitoringu jest istotny także podczas przesyłania danych o wykrytych zdarzeniach. Komunikacja taka odbywa się pomiędzy komponentami systemowymi i jest najczęściej w pełni zautomatyzowana. Informacje o zdarzeniach wysyłane są w małych porcjach z dużą częstotliwością. W związku z tym nie jest wymagana duża średnia szybkość bitowa w sieci, ale kluczowe stają się parametry takie jak opóźnienia sieciowe oraz zmienność opóźnienia. W sytuacjach krytycznych, przy dużej liczbie zdarzeń i przy wykorzystaniu zaawansowanych usług zbyt duże opóźnienia mogą wpłynąć na wynik analizy. Dodatkowo zapewnienie małych opóźnień otwiera drogę do stosowania reguł przetwarzania danych wrażliwych na subtelne zależności czasowe pomiędzy zdarzeniami, wykorzystywanych do wykrywania skomplikowanych wzorców zachowań i zależności. W trakcie działania systemu MobiWatch wyróżnić można 4 modele wykorzystania sieci. Każdy z tych modeli związany jest z innym scenariuszem działania oraz z inną klasą usług koniec-koniec oferowaną przez mechanizmy zapewnienia jakości. Do scenariuszy tych należy:

- Pozyskiwanie strumieni multimedialnych na żywo
- Odtwarzanie strumieni multimedialnych na żywo
- Odtwarzanie materiałów archiwalnych
- Przesyłanie informacji o zdarzeniach

Odwzorowania scenariuszy do odpowiednich klas usług prezentuje 8.

**Tablica 8.** Odwzorowania scenariuszy na klasy usług koniec-koniec dla aplikacji MobiWatch

| **Nazwa scenariusza** | **Klasa usług koniec-koniec** | **Typ połączenia** | **Protokół** |
|---|---|---|---|
| Pozyskiwanie strumieni multimedialnych na żywo | RT Interactive | planowane | UDP |
| Odtwarzanie strumieni multimedialnych na żywo | RT Interactive | na żądanie | UDP |
| Odtwarzanie materiałów archiwalnych | MM Streaming | na żądanie | UDP |
| Przesyłanie informacji o zdarzeniach | Low Latency Data | planowane | TCP |

## 3.4 Aplikacje zdalne nauczanie

### HD Video

HD Video jest zaawansowanym systemem telekonferencyjnym z przeznaczeniem do wykorzystania w zdalnej edukacji. Aplikacja jest wyposażona w moduł planowania, oceny i monitorowania procesu nauczania (CAA - Computer Aided

Assessment system). W aplikacji obsługiwane są dwa podstawowe scenariusze – zdalny wykład oraz zdalny wykład otwarty. W obu przypadkach wykorzystywane są połączenia planowane. W przypadku wykładu otwartego uczestniczyć w nim mogą wszyscy użytkownicy aplikacji Video HD. Rezerwacja połączeń odbywa się po zaplanowaniu wykładu za pomocą CAA, którego usługi także działają w sieci IPv6 QoS. W obu scenariuszach przesyłane są rodzaje danych wymienione w tabeli 9.

**Tablica 9.** Odwzorowanie scenariuszy na klasy usług koniec-koniec dla aplikacji zdalne nauczanie

| **Scenariusz - rodzaj danych** | **Klasa usług koniec-koniec** | **Protokół transportowy** | **Typ połączenia** | **Wymagana szybkość bitowa na poziomie aplikacji** |
|---|---|---|---|---|
| Usługi CAA | Standard | TCP | — | Brak wymagań |
| Strumień video | RT Interactive | UDP, RTP | Planowane | Maksymalna szybkość bitowa:10Mb/s |
| Strumień audio | Telephony | UDP, RTP | Planowane | Maksymalna szybkość bitowa: 44kb/s |
| Komunikaty aplikacji | Signalling | UDP / TCP | — | Wymagania do określenia na podstawie badań |

### Online Lab

Online Lab jest wirtualnym laboratorium obliczeniowym, wyposażonym w funkcjonalności dzielenia danych, kodu i wyników obliczeń. Użytkownik korzysta z wirtualnego pulpitu, pozwalającego definiować zadania obliczeniowe, publikować je dla innych użytkowników oraz wykonywać zdalnie, wykorzystując funkcje optymalizacji czasu odpowiedzi. Dostęp do aplikacji odbywa się przez graficzny interfejs WWW, co pozwala na jej wykorzystanie w zdalnym nauczaniu. Obliczenia wykonywane są przez dedykowane usługi obliczeniowe, działające w rozproszonym środowisku i podlegające stałemu monitorowaniu (pod względem zużycia zasobów – pamięci, procesora oraz dysku – oraz czasu odpowiedzi) przez moduł zarządzający. Podstawowy scenariusz zakłada zlecenie przez użytkownika wykonania zadania obliczeniowego, co powoduje wybór usługi obliczeniowej oraz zestawienie połączenia z usługą obliczeniową. Połączenia są planowane, bądź zestawiane na żądanie – w pierwszym przypadku moduł zarządzający dokonuje oceny zapotrzebowania na zasoby komunikacyjne, podejmując decyzję o parametrach połączeń na podstawie analizy strumienia zgłoszeń pochodzących od wszystkich użytkowników. W drugim przypadku, ocenie podlega pojedyncze zadanie obliczeniowe a wymagana szybkość bitowa dla połączenia zależy od rozmiaru danych i kodu skojarzonych z zadaniem oraz przewidywanego czasu obliczeń. W obu przypadkach kryterium jest czas oczekiwania użytkownika na realizację zadania. Tabela 10 przedstawia rodzaje danych wykorzystywane przez aplikację wraz z odpowiednimi klasami usług.

**Tablica 10.** Odwzorowanie scenariuszy na klasy usług koniec-koniec dla aplikacji Online Lab

| **Scenariusz - rodzaj danych** | **Klasa usług** | **Protokół transportowy** | **Typ połączenia** | **Wymagana średnia szybkość bitowa na poziomie aplikacji** |
|---|---|---|---|---|
| Dane i kod | High Throughput Data | TCP | Planowane/na żądanie | 1-10 Mb/s |
| Monitoring usług | Low Latency Data | TCP | Planowane | 64 kb/s |
| Komunikaty aplikacji | Signalling | TCP | — | Wymagania do określenia na podstawie badań |

## 4 Podsumowanie

W artykule zaproponowano odwzorowanie klas usług koniec-koniec na przykładowe aplikacje testowe w ramach Równoległego Internetu IPv6 QoS, a także odwzorowanie na klasy usług w sieci IP oraz sieci dostępowej WiFi. W porównaniu z dotychczas definiowanymi klasami usług dla aplikacji typu VoIP, IPTV, dla aplikacji Internetu Rzeczy definiujemy również klasę usług koniec-koniec Low Latency Data, która ma zagwarantować małe opóźnienia dla przekazu pakietów przez sieć dla aplikacji stosujących protokół TCP. Dla wszystkich klas usług w sieci IP zostały zaprojektowane i zaimplementowane mechanizmy QoS w urządzeniach umożliwiających wirtualizację opartych na platformach EZappliance (rutery brzegowe), NetFPGA i XEN (rutery szkieletowe). Urządzenia te współpracują również z systemem sygnalizacji opracowanym dla RI IPv6 QoS w oparciu o wytyczne ITU-T.

## Literatura

1. W. Burakowski, H. Tarasiuk, A. Beben, G. Danilewicz, „Virtualized network infrastructure supporting co-existence of Parallel Internet", In Proc. of SNPD 2012, Kyoto, August 2012.
2. H. Tarasiuk, et al., "Architektura i mechanizmy Równoległego Internetu IPv6 QoS", Przegląd Telekomunikacyjny, SIGMA NOT, 2011, nr 8-9, t. CD-ROM, s.944-954.
3. ITU-T Rec. Y.2001, "ITU-T, General overview of NGN", ITU Telecommunication Standardization Sector, December 2004.
4. S. Blake i in.," An Architecture for Differentiated Services", Internet RFC 2475, 10/1998.
5. H. Tarasiuk, J. Rogowski, "On the Signaling System in the IPv6 QoS Parallel Internet", In Proc. of CSNDSP 2012, Poznan 2012.
6. W. Burakowski, et al., "Provision of End-to-End QoS in Heterogeneous Multi-Domain Networks", Annals of Telecommunications, Springer Eds., Vol. 63, Issue 11, 2008.
7. J. Babiarz, et al., "Configuration Guidelines for DiffServ Service Classes", Internet RFC 4594, August 2006.

8. K. Chan, J. Babiarz, F. Baker, “Aggregation of Diffserv Service Classes”, Internet RFC 5127, February 2008.
9. P. Wiśniewski, et al., “Implementacja warstwy danych i sterowania dla Równoległego Internetu IPv6 QoS”, W materiałach konferencji KSTiT 2012, Warszawa 2012.
10. B. Lewandowski, C. Mazurek, A. Radziuk, „Aplikacja inteligentnego zarządzania energią w środowisku domowym jako usługa Internetu Przyszłości”, Konferencja i3 2010, Wrocław 2010.
11. L. Tan, N. Wang, “Future Internet: The Internet of Things”, In Proc. 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), pp. 376 - 380, 2010.
12. H. Tarasiuk, et al., „System sygnalizacji oraz scenariusze aplikacyjne w Równoległym Internecie IPv6 QoS”, KSTiT 2012, Warszawa 2012.
13. G. Wu, et al., “M2M: From mobile to embedded internet”, Communications Magazine, IEEE, vol. 49, no. 4, , Apr. 2011, pp. 36-43.
14. D. Mierzwinski, D. Walczak, M. Wolski, M. Wrzos, “Video and Sensor Data Integration in a Service-Oriented Surveillance System”, Scalable Computing: Practice and Experience, vol. 12, 2011.
15. D. Luckham, “The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems”, Addison-Wesley Professional, May 2002.

# Sieci świadome przekazywanej treści

Andrzej Bęben

Instytut Telekomunikacji, Politechnika Warszawska

**Streszczenie** Artykuł przedstawia sieci świadome przekazywanej treści (CAN ang. *Content Aware Networks*), które zostały zaprojektowane dla zapewnienia użytkownikom ujednoliconego dostępu do treści multimedialnych oraz efektywnego przekazu treści od serwera przechowującego replikę żądanej treści do użytkownika. Realizacja sieci treści wymaga opracowania nowej architektury sieci, która wspiera funkcje związane z publikowaniem treści, wyszukiwaniem treści oraz jej dostarczaniem. W artykule przedstawiono założenia, wymagania oraz przegląd proponowanych rozwiązań i kierunki badań dotyczących sieci świadomych przekazywanej treści.

## 1 Wprowadzenie

W ostatnich latach w sieci Internet można zaobserwować znaczący wzrost ruchu związanego z przekazem treści multimedialnych, takich jak filmy wideo, utwory muzyczne, transmisje wydarzeń kulturalnych i sportowych czy też programy telewizyjne i audycje radiowe. Zgodnie z wynikami badań przedstawionymi w [1], przekaz treści multimedialnych stanowi co najmniej 50% ruchu w sieci Internet. Jednocześnie, prognozy dotyczące ruchu w Internecie przewidują dalszy jego wzrost ze względu na rozpowszechnienie obrazów przygotowanych w wysokiej rozdzielczości (ang. *High Definition*) oraz obrazów trójwymiarowych (3D). Obecnie stosowany model dystrybucji treści zakłada wykorzystanie systemów pośredniczących, które umożliwiają przechowywanie publikowanej treści oraz jej udostępnianie na żądanie użytkowników. Systemy te obejmują m.in. dedykowane platformy dystrybucji treści przeznaczone dla danego usługodawcy, np. systemy "wideo na życzenie"VoD (ang. *Video on Demand*), powszechnie dostępne systemy udostępniania treści, takie jak YouTube, Flickr, czy też sieci wymiany danych typu peer-to-peer (p2p), umożliwiające bezpośredni przekaz zbiorów pomiędzy użytkownikami. Wspólną cechą powyższych rozwiązań jest zrealizowanie systemu dystrybucji treści na poziomie aplikacji i wykorzystanie sieci Internet jedynie do przekazu pakietów.

Główne ograniczenia obecnego modelu dystrybucji treści, zgodnie z [2], wynikają z: (1) braku jednolitego i powszechnego systemu identyfikatorów treści, który ogranicza dostępność opublikowanej treści jedynie do grona użytkowników danego systemu pośredniczącego, (2) ograniczonej współpracy pomiędzy operatorami systemów pośredniczących a operatorami sieci, co przyczynia się do nieefektywnego wykorzystania zasobów sieci i serwerów, w szczególności brak

informacji o przesyłanej treści powoduje, iż popularna treść jest przesyłana wielokrotnie powodując przeciążenia sieci, (3) braku odpowiednich mechanizmów wspierających przekaz treści, które umożliwiałyby zapewnienie wymaganej jakości przekazu pakietów, wykorzystanie transmisji rozsiewczej, przechowywanie replik popularnej treści w pamięci podręcznej węzłów, czy też dynamiczny wybór ścieżek pomiędzy serwerem treści a użytkownikiem umożliwiający równoważenie obciążenia sieci.

Powyższe ograniczenia, a także prognozowany dalszy wzrost ruchu związanego z przekazem treści multimedialnych, przyczyniły się do rozpoczęcia prac badawczych nad nową architekturą sieci, tzw. siecią świadomą przekazywanej treści - CAN (ang. *Content Aware Networks*), nazywanej również siecią zorientowaną na przekaz treści ICN (ang. *Information Centric Network*). Głównym celem sieci CAN jest zapewnienie ujednoliconego dostępu do publikowanej treści oraz zapewnienie efektywnego dostarczenia żądanej przez użytkownika treści od najlepszego źródła biorąc pod uwagę wymagania dotyczące jakości przekazu żądanej treści, warunki ruchowe panujące w sieci, a także obciążenie serwerów treści. Sieci CAN budowane są w oparciu o nowy model dystrybucji, którego istotą jest dostarczenie treści żądanej przez użytkownika natomiast nie jest istotny adres czy też fizyczna lokalizacja serwera treści obsługującego żądanie. Biorąc pod uwagę zjawiska obserwowane w obecnym Internecie, sieci CAN są powszechnie uważane za jeden z kluczowych elementów Internetu Przyszłości [2, 3].

Spośród rozwiązań proponowanych dla sieci treści należy wyróżnić rozwiązania ewolucyjne oraz rewolucyjne. Rozwiązania ewolucyjne zakładają rozszerzenie istniejącej infrastruktury sieci Internet o nowe węzły sterujące procesem wyszukiwania i pobierania treści. Węzły te są odpowiedzialne za wybór najlepszego serwera w oparciu o informacje dotyczące wymaganej jakości przekazu treści, aktualnego stanu serwerów oraz warunków ruchowych w sieci. Przykładami ewolucyjnych rozwiązań są sieci Content Delivery Networks (CDN) [4], sieci p2p [5] i p4p [6]. Z drugiej strony, rozwiązania rewolucyjne zakładają zastąpienie bądź radykalną zmianę istniejącej infrastruktury sieci przez wprowadzenie nowych węzłów zorientowanych na przekaz treści. W rozwiązaniach tych przyjęto nową organizację sieci, która umożliwia zastosowanie efektywnych algorytmów wyszukiwania treści. Przykładami rewolucyjnych rozwiązań są sieci CCN/NDN [7], DONA [8], COMET [9], czy też sieć PI CAN [10] opracowana w ramach projektu Inżynieria Internetu Przyszłości [11].

Należy podkreślić, że rozwiązania ewolucyjne, tj. sieci CDN, są już obecnie stosowane bądź w najbliższej przyszłości będą wdrażane w sieciach operatorów. Z tego względu, prace badawcze w głównej mierze dotyczą zwiększenia efektywności stosowanych mechanizmów i algorytmów oraz zapewnienia współpracy pomiędzy sieciami treści różnych operatorów [12]. W przypadku rozwiązań rewolucyjnych prace badawcze dotyczą opracowania prototypów, weryfikacji przyjętych rozwiązań oraz oceny ich efektywności.

Artykuł jest zorganizowany w następujący sposób. W rozdziale 2 przedstawiono podstawowe procesy realizowane w sieciach świadomych treści. W rozdziale 3 omówiono wymagania dotyczące stosowanych mechanizmów i algoryt-

mów. Przegląd ewolucyjnych i rewolucyjnych rozwiązań proponowanych dla sieci świadomych przekazywanej treści zmieszczono w rozdziale 4. Ostatecznie, rozdział 5 podsumowuje artykuł oraz przedstawia kierunki aktualnie prowadzonych prac badawczych.

## 2 Procesy w sieciach CAN

W sieciach świadomych przekazywanej treści wyróżniamy trzy podstawowe procesy związane z publikowaniem treści, wyszukiwaniem treści oraz jej dostarczaniem do użytkownika. Publikowanie treści jest inicjowane przez wydawcę treści, który chce udostępnić posiadane materiały użytkownikom. W tym celu wydawca umieszcza publikowaną treść na serwerach treści, a następnie rejestruje ją w sieci CAN. Kluczowym elementem procesu publikowania jest utworzenie replik treści oraz ich rozmieszczenie na serwerach treści w sposób proporcjonalny do zainteresowań użytkowników oraz możliwości obsługowych serwerów. Ponadto, jest niezbędnym nadanie jednoznacznego identyfikatora treści, który umożliwi użytkownikom jej wyszukanie i pobranie. Proces wyszukiwania treści jest inicjowany przez użytkownika w momencie wysłania żądania danej treści. Wyszukiwanie treści polega na uzyskaniu informacji o serwerach przechowujących żądaną treść lub jej repliki i wybraniu najlepszego serwera z punktu widzenia satysfakcji użytkownika oraz wykorzystania zasobów. Wybór serwera powinien uwzględniać obciążenie serwerów, wymagania dotyczące przekazu treści oraz warunki ruchowe w sieci. Kluczowym elementem procesu wyszukiwania są efektywne i skalowalne algorytmy wyszukiwania treści oraz mechanizmy zbierania informacji o stanie serwerów i sieci. Należy zwrócić uwagę, iż algorytmy te powinny umożliwić wyszukanie żądanej treści spośród wszystkich treści opublikowanych w sieci CAN. Ostatnim z procesów realizowanych w sieci CAN jest proces dostarczania treści. W przypadku rozwiązań ewolucyjnych przekaz treści jest realizowany z wykorzystaniem sieci Internet. W tym przypadku głównym ograniczeniem jest brak możliwości wyboru drogi, po której będzie przesyłana treść, oraz brak możliwości realizacji połączeń typu ąnycast"i "multicast". Z tego względu, w rozwiązaniach rewolucyjnych zdefiniowano nowe węzły sieci, które realizują funkcje wspierające przekaz treści, m.in. umożliwiają elastyczny wybór drogi, wspierają połączenia rozsiewcze, a także umożliwiają przechowywanie przesyłanej treści w pamięci podręcznej węzłów.

## 3 Wymagania dotyczące sieci CAN

W rozdziale przedstawiono wymagania dotyczące podstawowych funkcji realizowanych w sieciach świadomych przekazywanej treści. W szczególności przedstawiono wymagania dotyczące identyfikatorów treści, algorytmów wyszukiwania i rutingu, mechanizmów wspierających przekaz treści oraz kontrolę dostępu.

### 3.1 Identyfikatory treści

Identyfikator treści w sieciach świadomych treści umożliwia jednoznaczne zidentyfikowanie opublikowanych obiektów. Z tego względu, identyfikator musi być unikalny w całej sieci. Ponadto, identyfikator treści musi posiadać wystarczającą pojemność dla zaadresowania wszystkich obiektów udostępnionych w sieci oraz zapewnić możliwość elastycznego rozszerzenia przestrzeni adresowej w miarę wzrostu liczby opublikowanych obiektów. Dodatkowo, identyfikator treści powinien być niezależny od lokalizacji treści, dzięki czemu jest możliwe tworzenie nowych replik oraz usuwanie lub przenoszenie treści pomiędzy serwerami treści w odpowiedzi na zmieniające się zainteresowanie użytkowników.

Opracowanie powszechnego, efektywnego i skalowalnego schematu identyfikacji treści jest jednym z kluczowych wyzwań stojących przed projektantami sieci świadomych treści. Obecnie są rozważane dwa podejścia, tj. rozwiązania hierarchiczne oraz rozwiązania z identyfikatorami bez wyróżnionej struktury. Rozwiązanie hierarchiczne zakłada, że identyfikator treści jest zbudowany z dwóch części, tj. globalnie unikalnego identyfikatora wydawcy oraz lokalnego identyfikatora treści, nadanego przez wydawcę. Przykładem globalnych identyfikatorów wydawców mogą być nazwy domen, np. cnn.com, które są stosowane w przypadku dużych wydawców, bądź też adresy e-mail, stosowane w przypadku indywidualnych wydawców. Postać lokalnego identyfikatora jest zależna wyłącznie od wymagań danego wydawcy. W przypadku hierarchicznych identyfikatorów informacja o treści jest najczęściej przechowywana w rozproszonej bazie danych o strukturze drzewa. Przykładem takiej struktury mogą być system nazw DNS [13] (ang. *Domain Name System*), stosowany w obecnej sieci Internet, lub system identyfikatorów cyfrowych DOI [14] (ang. *Digital Object Identifier*). Drugie rozwiązanie proponowane dla systemu identyfikacji treści zakłada płaską strukturę identyfikatorów. W tym przypadku identyfikatorem jest ciąg binarny o określonej długości, w którym nie wyróżniono wewnętrznej struktury. Identyfikator jest najczęściej generowany przy pomocy funkcji skrótu, np. MD5, na podstawie binarnej zawartości obiektu. W przypadku treści, która nie jest jeszcze znana w chwili publikowania, np. transmisja 'na żywo" wydarzeń sportowych, identyfikator może być wyznaczony na podstawie algorytmu UUID [15]. Przedstawione algorytmy nie gwarantują, że wyznaczony identyfikator będzie unikalny. Z tego względu, jednoznaczność identyfikatora należy zweryfikować w chwili publikowania treści. W przypadku płaskich identyfikatorów, informacja o treści jest najczęściej przechowywana w strukturach bazujących na rozproszonych tablicach skrótów DHT (ang. *Distributed Hash Tables*). Struktury te są powszechnie stosowane w sieciach peer-to-peer [5], ponieważ umożliwiają efektywne wyszukiwanie elementów przy relatywnie dużych strukturach danych. Ponadto, płaskie identyfikatory umożliwiają efektywne wykorzystanie przestrzeni adresowej, pozwalając na utworzenie względnie krótkich identyfikatorów treści. Jednakże, brak wewnętrznej struktury identyfikatorów uniemożliwia ich agregowanie. Cecha ta jest niekorzystna z punktu widzenia rutingu, gdyż może istotnie przyczynić się do zwiększenia rozmiarów tablic kierowania pakietów. Ponadto, przydzielenie losowych identyfikatorów treści w znacznym stopniu utrudnia kontrolę nad miejscem

przechowywania informacji o publikowanej treści. W efekcie, informacja o treści opublikowanej na lokalnych serwerach może być przechowywana w odległej domenie, przyczyniając się do wydłużenia czasu wyszukiwania treści.

### 3.2 Algorytmy wyszukiwania i ruting

Kluczowym elementem sieci świadomych treści są algorytmy wyszukiwania treści oraz rutingu. Algorytmy te powinny zapewnić, że jeżeli żądana przez użytkownika treść jest dostępna w sieci to zostanie ona odnaleziona, a żądanie jej pobrania zostanie przekierowane do najlepszego serwera treści. Wybór serwera powinien uwzględniać wymagania dotyczące jakości przekazu treści, stan obciążenia serwerów oraz warunki ruchowe w sieci.

Obecnie projektowane sieci świadome treści zakładają wykorzystanie jednego z dwóch rozwiązań: 1) rozwiązanie wzorowane na systemie nazw DNS oraz 2) ruting na podstawie identyfikatorów treści (ang. *route by name*). Pierwsze rozwiązanie zakłada, że procesy wyszukiwania oraz dostarczania treści są realizowane w dwóch niezależnych fazach. W pierwszej fazie, algorytm wyszukiwania zbiera informacje o lokalizacji replik żądanej treści, obciążeniu serwerów i sieci, a także dostępnych drogach przekazu treści pomiędzy serwerami a użytkownikiem. Na tej podstawie algorytm podejmuje decyzję o wyborze najlepszego serwera, który będzie obsługiwał żądanie użytkownika. W drugiej fazie, użytkownik wysyła żądanie pobrania treści do wybranego przez sieć serwera treści, który rozpoczyna przekaz pakietów zgodnie ze ścieżką przekazu wybraną w pierwszej fazie. Rozdzielenie procesu wyszukiwania od wyznaczenia dróg przekazu treści pozwala na zastosowanie optymalnych struktur danych dla każdego procesu. Cecha ta jest szczególnie istotna z punktu widzenia algorytmów wyszukiwania, które ze względu na ogromną liczbę publikowanej treści wymagają efektywnych struktur przechowywania danych. Zastosowane algorytmy wyszukiwania są związane z przyjętym schematem identyfikatorów. W przypadku identyfikatorów hierarchicznych najczęściej są stosowane algorytmy wyszukiwania oparte na strukturach drzewa, które charakteryzują się złożonością na poziomie $O(h)$, gdzie $h$ jest wysokością drzewa. Większą efektywność algorytmów wyszukiwania można uzyskać w przypadku algorytmów opartych na rozproszonych tablicach skrótów bądź filtrach Bloom'a. Przykładami mogą być algorytmy wyszukiwania CAN [16] (ang. *Content-Addressable Network*), czy też COLOCAN [17] (ang. *COntent LOcalisation for CAN*).

Rozwiązania wykorzystujące ruting na podstawie identyfikatorów treści, przedstawione m.in. w [18, 19, 20], zakładają, że proces wyszukiwania oraz dostarczania treści jest realizowany w jednej fazie. W tym przypadku, żądania użytkowników są przesyłane przez węzły sieci do serwerów na podstawie identyfikatorów treści. Identyfikator treści pełni jednocześnie rolę adresu sieciowego. Proces wyszukiwania treści jest realizowany "węzeł po węźleąż do momentu znalezienia pierwszej repliki żądanej treści. Rozwiązania wykorzystujące ruting na podstawie identyfikatorów treści wymagają specjalnej organizacji sieci, aby zapewnić efektywne działanie algorytmów wyszukiwania. Najczęściej rozważaną topologią

sieci jest topologia drzewa, w której informacja o treści jest agregowana na danym poziomie drzewa, a następnie jest propagowana do węzła żodzica". Istotną cechą tych rozwiązań jest naturalne wykorzystanie pamięci podręcznej w węzłach sieci. W przypadku, gdy węzeł odbierze żądanie dotyczące treści umieszczonej w jego pamięci podręcznej, węzeł kończy proces wyszukiwania i rozpoczyna dostarczanie treści do użytkownika. Należy jednakże zwrócić uwagę, że proponowane rozwiązanie jest efektywne wyłącznie dla treści, która ma ograniczona populację i jest bardzo popularna. W takim przypadku, większość żądań zostanie obsłużona przez węzły pośredniczące.

### 3.3 Mechanizmy wspomagające przekaz treści

Jednym z głównych ograniczeń obecnej sieci Internet związanych z dystrybucją treści multimedialnych jest brak mechanizmów wpierających przekaz treści. Należy zwrócić uwagę, iż sieć Internet była projektowana dla zapewnienia komunikacji typu punkt-punkt, pomiędzy dwoma użytkownikami lub pomiędzy użytkownikiem a serwerem. W przypadku sieci świadomych treści podstawowym rodzajem połączeń są połączenia typu ąnycastóraz połączenia "multicast". Połączenia ąnycastśą podstawowym rodzajem połączeń, które są wykorzystywane w przypadku, gdy treść żądana przez użytkownika może być dostarczona przez dowolny serwer treści. Połączenia "multicastśą wykorzystywane w przypadku transmisji rozsiewczych, takich jak transmisja programów telewizyjnych lub audycji radiowych.

Istotnym wymaganiem sieci świadomej treści jest wykorzystanie pamięci podręcznej w węzłach sieci. Pamięć ta umożliwia przechowywanie replik przesyłanej treści, które mogą być wykorzystane dla zrealizowania przyszłych żądań użytkowników. Zgodnie z wynikami badań przedstawionymi w [21, 22, 23], zastosowanie pamięci podręcznej w węzłach sieci przyczynia się do znacznego zredukowania obciążenia serwerów oraz sieci. Jednakże niezbędnym jest opracowanie efektywnych algorytmów zarzadzania pamięcią podręczną, które zapewnią, że najbardziej popularna treść będzie zawsze dostępna w pamięci podręcznej.

### 3.4 Mechanizmy kontroli dostępu

Biorąc pod uwagę, iż zdecydowana większość publikowanych treści jest objęta prawami autorskimi, ograniczeniami dotyczącymi dystrybucji itp., w sieci świadomej treści niezbędnym jest zastosowanie mechanizmów kontroli dostępu do treści. Mechanizmy te powinny zapewnić autoryzację użytkownika, weryfikację jego uprawnień oraz kontrolę wykorzystanych zasobów. Ponadto, kontrola dostępu jest również konieczna w przypadku procesu publikowania treści, aby utworzenie, modyfikacja bądź usunięcie opublikowanej treści było możliwe wyłącznie przez uprawnione do tego osoby.

Dodatkowym wymaganiem jest konieczność zapewnienia wiarygodności dostarczanych treści. Należy zwrócić uwagę, iż w sieciach CAN treść jest dostarczana z dowolnego serwera. Z tego względu, użytkownik nie ma możliwości zweryfikowania wiarygodności otrzymanej treści na podstawie adresu serwera. Za-

tem niezbędnym jest wprowadzenie dodatkowego mechanizmu umożliwiającego zweryfikowanie wiarygodności pobranej treści.

## 4 Przykładowe rozwiązania

W rozdziale tym przedstawiono przykładowe rozwiązania dotyczące sieci świadomych przekazywanej treści, obejmujące zarówno rozwiązania ewolucyjne bazujące na rozszerzeniu obecnej sieci Internet, jak i rozwiązania rewolucyjne wymagające istotnych zmian w infrastrukturze sieci.

### 4.1 Sieci CDN

Sieci CDN (ang. *Content Delivery Networks*) [4, 24] zaliczamy do ewolucyjnych rozwiązań, które wspomagają dystrybucję treści w sieci Internet. Zasada działania sieci CDN jest następująca:

- Operator sieci CDN umieszcza w sieci Internet pewną liczbę serwerów replik treści (ang. *surrogate servers*), które będą przechowywać i udostępniać użytkownikom materiały dostarczone przez wydawców. Serwery replik są umieszczane blisko użytkowników, zwykle w punktach koncentracji ruchu, tak aby zapewnić efektywny przekaz treści.
- Wydawca treści zamawia usługę dostarczania treści w sieci CDN i umieszcza treść na swoim serwerze (ang. *origin server*).
- Algorytmy zarządzania treścią w sieci CDN tworzą repliki materiałów udostępnionych przez dostawcę i umieszczają je na serwerach replik zgodnie z prognozowanym zainteresowaniem użytkowników. Jednocześnie, uaktualniają tablice przekierowania żądań użytkowników.
- Żądania generowane przez użytkowników są przekierowywane do najbliższego serwera replik, np. za pomocą mechanizmów DNS lub HTTP redirect.

Sieci CDN są wykorzystywane do dystrybucji treści, o której wydawca wie *a'priori*, że będzie popularna, np. aktualizacje oprogramowania, popularne serwisy internetowe itd.

### 4.2 Sieć DONA/CURLING

Sieć DONA [7] (ang. *Data Oriented Architecture*) oraz CURLING [19] należą do rozwiązań rewolucyjnych, w których zastosowano ideę rutingu na podstawie identyfikatorów treści. W obu rozwiązaniach założono, że sieć świadoma treści jest zbudowana w postaci drzewa. Użytkownicy, a także serwery treści są umieszczone w domenach stanowiących liście drzewa. Informacja o opublikowanej treści jest przechowywana w domenie, w której jest umieszczony serwer treści, następnie zaś jest ona propagowana w kierunku korzenia drzewa. Algorytm wyszukiwania działa następująco. Jeżeli domena, w której odebrano żądanie od użytkownika posiada informacje o żądanej treści, to żądanie zostaje przekierowane do odpowiedniego serwera treści. Natomiast w przeciwnym przypadku,

żądanie zostaje przekierowane do domeny żodzica". Proces jest kontynuowany aż do momentu odnalezienia żądanej treści lub osiągnięcia "korzenia"drzewa. Jeżeli "korzeń"drzewa nie dysponuje informacją o żądanej treści, oznacza to, że dana treść nie jest dostępna w sieci.

### 4.3 CCN/NDN

Sieci CCN /NDN [7, 25] (ang. *Content Centric Network / Named Data Networking*) należą do rozwiązań rewolucyjnych, w których podstawowym założeniem jest wykorzystanie pamięci podręcznej we wszystkich węzłach sieci. Ponadto, pakiety są kierowane na podstawie identyfikatorów treści. Istotną cechą proponowanego rozwiązania jest zastosowanie samo certyfikujących się identyfikatorów treści. Działanie sieci jest następujące. Użytkownik, który chce pobrać daną teść wysyła żądanie, które zawiera identyfikator treści. Pakiet ten jest przesyłany przez węzły sieci w kierunku do serwera przechowującego żądaną treść. W przypadku, gdy w pamięci podręcznej węzła pośredniczącego jest dostępna żądana treść, zostanie ona przesłana do użytkownika tą samą drogą jaką był przesłany pakiet z żądaniem treści. Jednocześnie wszystkie węzły pośredniczące umieszczają przesyłaną treść w swojej pamięci podręcznej. Należy zwrócić uwagę, iż w sieci CCN/NDN dana treść jest przesyłana jednokrotnie, a zarazem możliwe jest spełnienie wszystkich kolejnych docierających do węzłów pośredniczących żądań dotyczących tej samej treści.

### 4.4 Sieć COMET

Sieć COMET (ang. *Content Mediator Architecture for Content-aware Networks*) została opracowana w ramach projektu 7 Programu Ramowego UE [9, 26, 27]. Główna idea tej sieci polega na wprowadzeniu warstwy mediacji, która umożliwia optymalny wybór serwera i ścieżki dostarczania treści. Decyzja jest podejmowana na podstawie zgromadzonej informacji o: (1) lokalizacji replik żądanej przez użytkowników treści, (2) aktualnym obciążeniu serwerów treści, (3) dostępnych ścieżkach dostarczania treści oraz ich obciążenia. W tym celu zaproponowano algorytm decyzyjny bazujący na wielokryterialnej teorii decyzji, przedstawiony w [28]. Wyniki dotyczące efektywności sieci COMET wskazują, że proponowane mechanizmy mediacji pozwalają istotnie zwiększyć efektywność przekazu treści dzięki równoważeniu obciążenia serwerów oraz sieci.

### 4.5 Sieć PI CAN

Sieć PI CAN [10, 29] (ang. *Parallel Internet Content Aware Network*) opracowano w ramach projektu Inżynieria Internetu Przyszłości [11]. Proponowane rozwiązanie obejmuje: (1) ujednolicony dostęp do treści, (2) nowy schemat identyfikatorów treści oraz metodę przechowywania i wyszukiwania informacji o treści bazującą na rozproszonych tablicach skrótów, (3) mechanizmy sterowania siecią i pozyskiwania informacji o obciążeniu serwerów i warunkach ruchowych

w sieci, (4) proces decyzyjny umożliwiający wybór optymalnego serwera i ścieżki w sieci oraz (5) skalowalną metodę przekazu pakietów umożliwiającą wybór ścieżki w sieci dla każdego żądania. Istotną cechą opracowanego rozwiązania jest realizacja sieci PI CAN na urządzeniach umożliwiających wirtualizację.

## 5 Podsumowanie

Sieci świadome przekazywanej treści są aktualnie przedmiotem badań prowadzonych przez zespoły naukowo-badawcze z całego świata. Niniejszy artykuł prezentuje ogólną koncepcję oraz wymagania odnoszące się do tego typu sieci, jak również zaproponowane dotychczas rozwiązania dla ich realizacji. Mimo że propozycje te różnią się od siebie przez wzgląd na zastosowane mechanizmy i algorytmy, to niewątpliwie każda z nich zgodna jest z podstawowym założeniem wiążącym się z sieciami świadomymi przekazywanej treści: głównym paradygmatem dotyczącym sieci jest dostarczanie treści użytkownikom, nie zaś zapewnienie komunikacji pomiędzy dwoma urządzeniami dołączonymi do sieci, jak to jest w przypadku obecnego Internetu.

Biorąc pod uwagę aktualny stan prac możemy przyjąć, iż ogólne założenia dotyczące sieci CAN zostały już sformułowane. Jednakże nadal w obszarze tym prowadzone są intensywne prace badawcze, które dotyczą, między innymi, takich kwestii jak: modele analityczne opisujące sieci CAN (zagadnienie to ma szczególne znaczenie w aspekcie badań dotyczących skalowalności proponowanych rozwiązań), analiza pojemności sieci CAN, zagadnienia odnoszące się do wykorzystania pamięci podręcznej w węzłach sieci (zarządzanie pamięcią podręczną, algorytmy relokacji treści przechowywanych w pamięciach podręcznych itp.), bezpieczeństwo w sieciach CAN, wsparcie dla mobilności, jak również możliwość współdziałania poszczególnych rozwiązań CAN ze sobą oraz z innymi sieciami (w tym z obecnymi sieciami opartymi o stos protokołów TCP/IP).

## Literatura

1. C. Labovitz, et al., "ATLAS Internet Observatory 2009 Anual Report", dostępny na stronie `http://www.nanog.org/meetings/nanog47/presentations/Monday/Labovitz_ObserveReport_N47_Mon.pdf`
2. EC FIArch Group, Fundamental Limitations of Current Internet and the path to Future Internet, report ver. 0.9, 2010, dostępny na stronie `http://www.future-internet.eu/uploads/media/FIArch_Current_Internet_Limitations_V0_9.pdf`
3. Subharthi Paul, Jianli Pan, and Raj Jain, "Architectures for the Future Networks and the Next Generation Internet: A Survey", Computer Communications, Volume 34, Issue 1, 15 January 2011, Pages 2-42
4. M. Pathan, R. Buyya, "A Taxonomy and Survey of Content Delivery Networks", Technical Report, GRIDS-TR-2007-4, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, February, 2007.
5. Keong Lua, et al., "A survey and comparison of peer-to-peer overlay network schemes", IEEE Communications Surveys & Tutorials, vol.7, no.2, pp. 72- 93, 2005.

6. H. Xie, et al., "P4P: Provider Portal for Applications", ACM SIGCOMM 2008, Aug. 2008
7. V. Jacobson, et al., "Networking Named Content", CoNEXT 2009, Italy, December 2009, pp. 1-12.
8. T. Koponen, et al., "A Data-oriented (and Beyond) Network Architecture", Proc. ACM SIGCOMM '07, Kyoto, Japan, August 2007, pp. 181-92.
9. G. García, et al., "COMET: Content mediator architecture for content-aware networks", Future Network & Mobile Summit 2011, 15 - 17 June 2011, Warsaw, Poland
10. A. Beben, et al., "Content Aware Network based on Virtual Infrastructure", Proc. of 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2012), Kyoto, Japan, August 2012
11. W. Burakowski, H. Tarasiuk, A. Bęben, "Architektura Systemu IIP", Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011)
12. L. Peterson, B. Davie, "Framework for CDN Interconnection", IETF draft-ietf-cdni-framework-01, July 2012
13. Jaeyeon Jung, et al., "DNS Performance and the Effectiveness of Caching", IEEE/ACM Transactions on Networking (TON) archive, Volume 10 Issue 5, October 2002 Pages 589 - 603
14. S. Sun, L. Lannom, B. Boesch, "Handle System Overview", IETF RFC 3650, November 2003
15. P. Leach, M. Mealling, R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", IETF RFC 4122, July 2005
16. S. Ratnasamy, et al., "A scalable content-addressable network", SIGCOMM Comput. Commun. Rev. 31, 4 , August 2001, pp. 161-172.
17. P. Pecka, M. Nowak, S. Nowak, "Content localization for non-overlay content-aware networks", The 11th Int. Conf on Next generation Wired/Wireless Networks NEW2AN 2011, LNCS, Springer-Verlag 2011.
18. L. Zhang, et al., "Named Data Networking (NDN) Project", PARC Technical Report NDN-0001, 2010
19. W. K. Chai, et al., "CURLING: Content-ubiquitous resolution and delivery infrastructure for next generation services", IEEE Communications Magazine, vol. 49, no. 3, 2011, pp. 112-120
20. P. Jokela, et al., "LIPSIN: Line Speed Publish/Subscribe Inter-networking", Proc. ACM SIGCOMM '09, Barcelona, Spain, August 2009.
21. Ioannis Psaras, Wei Koong Chai and George Pavlou, "Probabilistic In-Network Caching for Information-Centric Networks", ACM SIGCOMM Workshop on Information-Centric Networks, Helsinki, Finland, August 2012
22. Wei Koong Chai, Ioannis Psaras and George Pavlou, "Cache Less for More In Information-Centric Networks", IFIP NETWORKING 2012, Prague, Czech Republic, May 2012.
23. I. Psaras et al., "Modelling and Evaluation of CCN-caching Trees", IFIP NETWORKING 2011, Valencia, Spain, May 2011
24. Cheng Huang, et al., "Measuring and evaluating large-scale CDNs", In Proceedings of the 8th ACM SIGCOMM conference on Internet measurement (IMC '08), ACM, New York, NY, USA, 2008, pp. 15-29
25. V. Jacobson, et al., "Networking named content", Proc. of the 5th international conference on Emerging networking experiments and technologies, 2009
26. A. Bęben, et al., "The content mediator architecture for content-aware networks", Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Wrzesień 2012

27. W. K. Chai, et al., "Final specification of mechanisms, protocols and algorithms for the content mediation system", COMET Deliverable D3.2, January 2012.
28. A. Bęben, et al., "Multi-criteria decision algorithms for efficient content delivery in content networks", Annals of Telecommunications, Special Issue on Networked digital media, Springer, 2012
29. A. Bęben i in., "Architektura sieci świadomych treści w systemie IIP", Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011)

# Realizacja sieci świadomej treści w oparciu o wirtualne zasoby Systemu IIP

Piotr Krawiec[1], Andrzej Bęben[2], Piotr Wiśniewski[2], Jordi Mongay Batalla[1], Jakub Borzdyński[3], Jakub Gutkowski[3], Piotr Ostapowicz[3], Andrzej Kozik[4], Radosław Rudek[4]

[1] Instytut Łączności – Państwowy Instytut Badawczy
[2] Instytut Telekomunikacji, Politechnika Warszawska
[3] Poznańskie Centrum Superkomputerowo-Sieciowe
[4] Politechnika Wrocławska

**Streszczenie** Artykuł przedstawia sposób realizacji sieci świadomej przekazywanej treści PI CAN (ang. *Parallel Internet Content Aware Network*) w oparciu o wirtualne zasoby udostępnione przez System IIP. Sieć PI CAN stanowi jeden z Równoległych Internetów, realizowanych na wspólnej infrastrukturze fizycznej Systemu IIP.

## 1 Wprowadzenie

Proces dystrybucji treści multimedialnych jest odpowiedzialny za największą część ruchu przekazywanego obecnie w sieci Internet, dodatkowo przewiduje się, iż ilość generowanego ruchu multimedialnego wzrośnie znacząco w najbliższej przyszłości. Jednakże obecna architektura Internetu oraz zastosowane technologie sieciowe nie są przystosowane do wydajnego i elastycznego przekazu danych multimedialnych. Dlatego też w ostatnich latach opracowana została koncepcja sieci świadomej przekazywanej treści CAN (ang. *Content Aware Network*), która ma na celu efektywną obsługę ustawicznie zwiększającego się wolumenu treści multimedialnych transmitowanych poprzez sieć Internet. Koncepcja ta przyjmuje, że elementarną właściwością sieci jest "świadomość" przesyłanych przez tą sieć treści. Świadomość ta objawia się m.in. poprzez następujące cechy: 1) sieć posiada zdolność wyboru serwera, na którym umieszczona jest kopia treści żądanej przez użytkownika, oraz 2) sieć ma możliwość wyboru ścieżki, poprzez którą transmisja żądanej treści odbędzie się w sposób optymalny z punktu widzenia efektywności i jakości przekazu. Prowadzone w tym obszarze prace badawcze zaowocowały kilkoma propozycjami architektur realizujących koncepcję sieci CAN, takimi jak Content-Centric Networking CCN [1] oraz Named Data Networking NDN [2] opracowywane pod kierownictwem Van Jacobsona z centrum badawczego PARC, DONA [3] zaproponowana przez zespół badawczy z Uniwersytetu Kalifornijskiego w Berkeley, czy NetInf [4] [5] opracowany w ramach projektów FP7 4WARD [6] i SAIL [7].

Koncepcja architektury dla sieci CAN bazującej na wirtualnej infrastrukturze sieciowej, została opracowana w ramach projektu Inżynieria Internetu Przyszłości (IIP) [8]. Zadaniem projektu było stworzenie innowacyjnego systemu dla

Internetu Przyszłości (ang. *Future Internet*), nazwanego Systemem IIP, bazującego na wirtualizacji infrastruktury sieciowej [9]. W oparciu o udostępnioną przez System IIP wirtualną infrastrukturę tworzone są tzw. Równoległe Internety, działające w pełnej izolacji od siebie i cechujące się odmienną płaszczyzną sterowania i przekazu danych. Jako jeden z trzech Równoległych Internetów realizowanych w projekcie opracowana została sieć PI CAN (ang. *Parallel Internet CAN*). Jej głównym zadaniem jest przekaz treści multimedialnych do użytkowników w efektywny sposób, a więc przy uwzględnieniu: 1) wymagań danej treści na jakość przekazu QoS (ang. *Quality of Service*), 2) lokalizacji użytkowników oraz serwerów na których przechowywana jest kopia żądanej treści, 3) aktualnego obciążenia tych serwerów jak również 4) aktualnych warunków ruchowych panujące w sieci.

Niniejszy artykuł przedstawia system PI CAN, stanowiący implementację zaproponowanej architektury sieci świadomej przekazywanej treści. System ten został stworzony na bazie urządzeń umożliwiających wirtualizację zasobów sieciowych zastosowanych do budowy Systemu IIP, takich jak: platforma Xen, będąca przykładem realizacji programowej, oraz karty NetFPGA i programowalne procesory sieciowe EZappliance, będące demonstratorem realizacji sprzętowej.

Zawartość artykułu przedstawia się następująco. Rozdział drugi prezentuje podstawowe założenia architektury sieci PI CAN, zaś rozdział trzeci opisuje podstawowe procesy wyszukiwania i pobierania treści przez użytkownika. W rozdziale czwartym przedstawiono sposób realizacji płaszczyzny sterowania w systemie PI CAN, podczas gdy realizacja płaszczyzny przekazu danych zaprezentowana jest w rozdziale piątym. Artykuł kończy podsumowanie.

## 2 Architektura sieci PI CAN

Architektura Równoległego Internetu CAN została zaprezentowana w sposób szczegółowy w artykułach [10] i [11]. Obejmuje ona trzy płaszczyzny, które grupują procesy odbywające się w zbliżonej skali czasowej.

Pierwsza z płaszczyzn, nazwana płaszczyzną zarządzania, obejmuje operacje przeprowadzane w długiej skali czasowej. Zaliczamy do nich działania związane z zarządzaniem treścią, wykonywane przez dostawców treści, takie jak: publikacja treści w sieci PI CAN, jej modyfikacja oraz usuwanie wcześniej opublikowanej treści. Druga grupa obejmuje działania przeprowadzane przez operatorów infrastruktury, jak np. zarządzanie ścieżkami przekazu treści, zarządzanie siecią (konfiguracja urządzeń, wymiarowanie, obsługa awarii), monitorowanie i pomiary, jak również kontrolowane przez sieć kopiowanie treści na wybrane serwery treści oraz zarządzanie serwerami treści.

Druga z płaszczyzn, zwana płaszczyzną sterowania, grupuje operacje związane z obsługą połączeń, które są inicjowane poprzez wygenerowanie przez użytkowników żądania pobrania treści. Płaszczyzna ta obejmuje następujące działania: obsługę żądania wysłanego przez użytkownika, uzyskanie informacji o lokalizacji kopii żądanej przez użytkownika treści i obciążeniu serwerów zawierających daną treść oraz wyszukanie informacji o dostępnych ścieżkach i warunkach

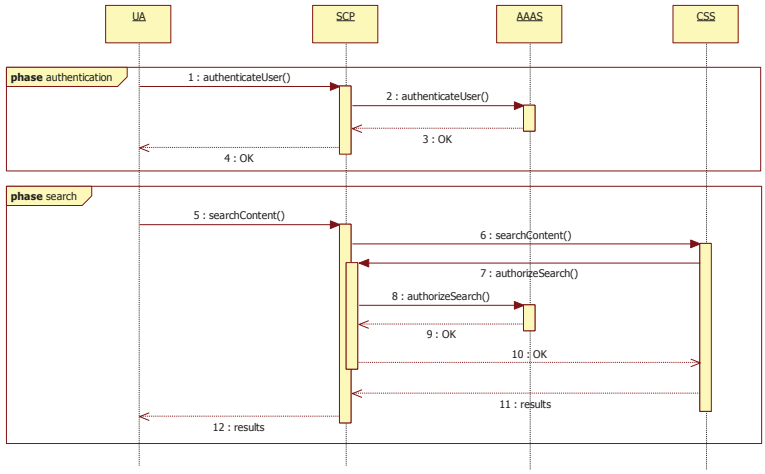
ruchowych panujących w sieci. Następnie, wykorzystując otrzymane dane, wykonywany jest algorytm decyzyjny związany z wyborem optymalnego serwera i ścieżki, na koniec zaś następuje konfiguracja wybranej ścieżki przekazu od serwera treści do użytkownika.

Ostatnia z płaszczyzn, tzw. płaszczyzna przekazu danych, odnosi się do operacji związanych z obsługą pojedynczych jednostek danych w węzłach sieci, a zatem wykonywanych w bardzo krótkiej skali czasowej. Działania przeprowadzane w ramach tej płaszczyzny obejmują przekaz pakietów z treścią, transmitowanych od serwera treści do użytkownika, jak również obsługę wiadomości generowanych i przesyłanych w ramach płaszczyzny zarządzania oraz płaszczyzny sterowania. Przekaz pakietów z treścią realizowany jest za pomocą mechanizmu dedykowanego dla sieci PI CAN [12] umożliwiającego ruting źródłowy. Oznacza to, iż węzły sieci podejmują decyzje dotyczące wyboru interfejsu, na który zostanie przesłany pakiet treści, na podstawie listy reguł przekazu przenoszonej w nagłówku danego pakietu. Dla przekazu wiadomości zarządzających i sterujących wykorzystano natomiast standardowy protokół IPv6.

Główne procesy występujące w sieci PI CAN to: 1) realizowany przez dostawców treści proces publikowania treści w sieci w celu jej udostępnienia użytkownikom, oraz 2) uruchamiany przez użytkowników proces pobierania treści. W celu przeprowadzenia powyższych procesów konieczne jest współdziałanie sieci PI CAN z następującymi elementami zewnętrznymi:

- Serwerem treści (ang. *Content Server*, CS), którego zadaniem jest przechowywanie kopii treści multimedialnych i strumieniowanie ich do użytkowników;
- Serwerem wyszukiwania treści (ang. *Content Search Server*, CSS), który odpowiedzialny jest za przechowywanie i wyszukiwanie metadanych opisujących daną treść; rekordy przechowywane w serwerze CSS zawierają m.in. takie pola, jak: rodzaj treści (audio, wideo), tytuł, wykonawca, wydawca, rok produkcji;
- Serwerem uwierzytelnienia i autoryzacji (ang. *Authentication, Autorization, Accounting Server*, AAAS), który wykonuje operacje związane z kontrolą dostępu do zasobów sieci PI CAN, takie jak uwierzytelnianie dostawców treści i użytkowników oraz autoryzacja podejmowanych przez nich działań.

System PI CAN zaprojektowano w taki sposób, aby umożliwiał on tworzenie odseparowanych logicznie sieci wirtualnych vCAN (ang. *virtual CAN*), z których każda może korzystać z własnego zbioru wymienionych wyżej elementów zewnętrznych. Serwery CS, CSS i AAAS dedykowane dla danej sieci wirtualnej mogą charakteryzować się funkcjonalnością całkowicie odmienną od serwerów wykorzystywanych w innych sieciach vCAN, uzależnioną od tego, jaki rodzaj treści i w jaki sposób będzie udostępniany w ramach danego vCAN. Dla przykładu, w ramach projektu IIP stworzono trzy sieci: 1) vCAN przeznaczony do obsługi treści multimedialnych (video i audio), 2) vCAN dla udostępniania treści medycznych, oraz 3) vCAN wykorzystany do stworzenia rozproszonego wirtualnego muzeum 3D.




**Rysunek 1.** Proces wyszukiwania treści

## 3 Proces konsumpcji treści

Elementarnym procesem związanym z działaniem sieci PI CAN jest proces konsumpcji treści przez użytkownika. Obejmuje on dwie fazy. W pierwszej z nich następuje wyszukanie w serwerze CSS metadanych treści odpowiadających opisowi przesłanemu przez użytkownika. Druga faza związana jest z pobraniem z sieci PI CAN treści wybranej przez użytkownika.

Poniżej zaprezentowano przykładowe scenariusze wymiany wiadomości sygnalizacyjnych dla procesu wyszukania oraz pobrania treści opublikowanej w sieci PI CAN. Należy zauważyć, że przeprowadzony przez dostawcę treści proces publikacji sprawia, iż w sieci PI CAN przechowywany jest odpowiedni rekord treści (ang. *content record*), zawierający m.in. identyfikator treści *contentId* oraz informacje o lokalizacji kopii danej treści na serwerach treści, zaś Serwer Wyszukiwania Treści posiada rekord wyszukiwania z metadanymi opisującymi daną treść.

Rys. 1 przedstawia diagram sekwencji wymiany wiadomości sygnalizacyjnych dla procesu wyszukiwania treści w serwerze CSS. Pierwsza faza polega na uwierzytelnieniu użytkownika poprzez wywołanie zdalnej metody *authenticateUser()* (krok 1 i 2), której parametrami jest nazwa użytkownika i jego hasło. W odpowiedzi użytkownik uzyskuje od serwera AAAS znacznik uwierzytelniający (*token*) (krok 3 i 4), który jest następnie wykorzystywany w celu autoryzacji operacji wyszukiwania i pobierania treści.

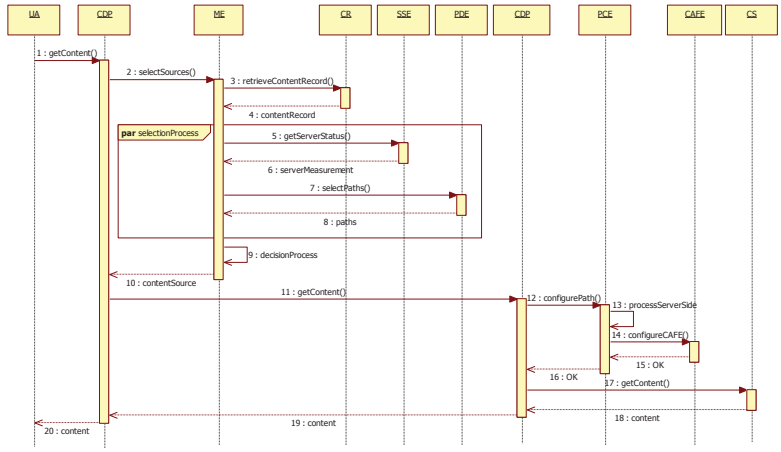
Jak możemy zauważyć, komunikacja pomiędzy siecią PI CAN a jednostkami zewnętrznymi (aplikacją użytkownika UA, serwerem AAAS i serwerem CSS),

która związana jest z płaszczyzną sterowania, odbywa się przy wykorzystaniu elementu pośredniczącego SCP (ang. *Service Control Proxy*). Wprowadzenie modułu SCP, oprócz zapewnienia obsługi sygnalizacji użytkownika, ma na celu zagwarantowanie separacji pomiędzy siecią PI CAN sieciami dostępowymi. W ten sposób sieć PI CAN może korzystać z własnej adresacji, niezależnej od adresacji stosowanej w komunikujących się z nią sieciach dostępowych. Dodatkowo, zastosowanie modułu SCP umożliwia wprowadzenie logicznej separacji wirtualnych sieci vCAN - SCP przekazuje żądania generowane przez użytkowników poszczególnych sieci vCAN do przynależnych do danej sieci wirtualnej serwerów CSS i AAAS. Mimo że przyjęty schemat działania zakłada, iż wszystkie przekazywane wiadomości sygnalizacyjne muszą zostać obsłużone przez SCP, moduł ten nie stanowi ograniczenia z punktu widzenia skalowalności systemu PI CAN, gdyż każde z napływających żądań użytkowników obsługiwane jest niezależnie. Tym samym, w przypadku zwiększania się liczby użytkowników sieci, obsługa wiadomości sygnalizacyjnych z zachowaniem odpowiedniej jakości może być osiągnięta poprzez wprowadzenie do sieci kolejnego SCP i przekierowania do niego części z generowanych żądań.

Po uwierzytelnieniu użytkownik przeprowadza proces wyszukania interesującej go treści poprzez wywołanie zdalnej metody *searchContent()* (krok 5), której parametrami są znacznik autoryzacyjny oraz opis poszukiwanej treści (zawierający np. tytuł, rodzaj lub tematykę treści). Powyższe dane są przekazywane przez moduł SCP do serwera CSS odpowiedniego dla danej sieci vCAN (krok 6). W sytuacji kiedy dana sieci wirtualna wymaga autoryzacji operacji wyszukiwania przez użytkownika, serwer CSS wywołuje zdalną metodę *autorizeSearch()* w celu przesłania znacznika autoryzacyjnego użytkownika do serwera AAAS (krok 7 i 8). Po uzyskaniu pozytywnego wyniku autoryzacji (krok 9 i 10), serwer CSS zwraca do modułu SCP rekordy zawierające metadane odpowiadające zdefiniowanemu przez użytkownika opisowi treści (krok 11). Zadaniem modułu SCP jest dodanie do każdego z otrzymanych rekordów wyszukiwania adresu URL (ang. *Uniform Resource Locator*), na który użytkownik może wysłać żądanie pobrania danej treści. Rekordy uzupełnione o adresy URL zwracane są następnie do aplikacji użytkownika (krok 12).

Adres URL, wykorzystywany przez użytkownika do pobrania treści, składa się z nazwy hosta (bądź adresu IP) modułu CDP (ang. *Content Delivery Proxy*) oraz identyfikatora treści contentId. Rola modułu CDP w fazie pobierania treści jest podobna do roli modułu SCP w procesie wyszukiwania - stanowi on element pośredniczący, wprowadzając separację pomiędzy siecią PI CAN a elementami zewnętrznymi, czyli aplikacją użytkownika i serwerem treści.

Diagram sekwencyjny przedstawiający proces pobierania treści przedstawia rys. 2. Użytkownik chcąc pobrać daną treść wysyła żądanie do modułu CDP wskazanego w adresie URL otrzymanym w rekordzie wyszukiwania (krok 1). W przypadku, kiedy vCAN, z którego użytkownik pobiera treść, posiada mechanizmy kontroli dostępu do treści, do wysyłanego żądania pobrania treści użytkownik dołącza własny znacznik autoryzacyjny.




**Rysunek 2.** Proces pobierania treści

Kiedy moduł CDP odbierze żądanie pobrania treści, przekazuje do modułu ME (ang. *Mediation Entity*) identyfikator *contentId* poszukiwanej treści w celu wskazania optymalnego serwera dla obsługi żądania użytkownika (krok 2). Wykorzystując otrzymany identyfikator moduł ME pobiera z rejestru CR (ang. *Content Registry*) rekord treści, który zawiera m.in. informację o lokalizacji wszystkich serwerów przechowujących kopię danej treści (krok 3 i 4). Następnie moduł ME odpytuje moduł SSE (ang. *Server Status Entity*) oraz moduł PDE (ang. *Path Discovery Entity*) w celu uzyskania informacji o obciążeniu serwerów treści (krok 5 i 6) oraz dostępnych w sieci ścieżkach pomiędzy serwerami a użytkownikiem (krok 7 i 8).

Moduł SSE odpowiedzialny jest za zbieranie i dostarczanie na żądanie informacji o aktualnym obciążeniu serwerów treści dołączonych do domeny PI CAN, w której uruchomiona jest instancja modułu. Dane dotyczące obciążenia serwera przekazywane są periodycznie do SSE przez zainstalowany na serwerze moduł agenta monitorującego MA (ang. *Monitoring Agent*).

Moduł PDE odpowiedzialny jest za zbieranie i dostarczanie na żądanie informacji o dostępnych w sieci ścieżkach między serwerami treści a konsumentami treści dla określonych klas usług. W każdej domenie sieci PI CAN występuje jedna instancja modułu PDE, z którą komunikuje się moduł RAE (ang. *Routing Awareness Entity*) znajdujący się w tej samej domenie oraz moduły ME. W przypadku modułów ME, PDE otrzymuje żądania zarówno od instancji ME znajdującej się w tej samej domenie, jak również z instancji znajdujących się w innych domenach. Informacje o ścieżkach przekazu treści dostarczane są do modułu PDE przez znajdujące się tej samej domenie moduły RAE i ME. W pierwszym przypadku są to ścieżki wychodzące z domeny, natomiast w drugim ścieżki zdalne.

Ścieżki zdalne, w przypadku których konsument treści znajduje się w tej samej domenie co rozpatrywane PDE a serwer w domenie odległej, są przechowywane w pamięci podręcznej modułu PDE w celu przyspieszenia działania procesu decyzyjnego i ograniczenia zapytań wysyłanych przez ME do innych domen.

Wykorzystując dane otrzymane z modułów SSE i PDE, moduł ME wykonuje algorytm decyzyjny (krok 9) mający na celu wybranie najlepszego serwera treści CS oraz najlepszej ścieżki pomiędzy wybranym serwerem a użytkownikiem. Wynik algorytmu jest zwracany do modułu CDP (krok 10).

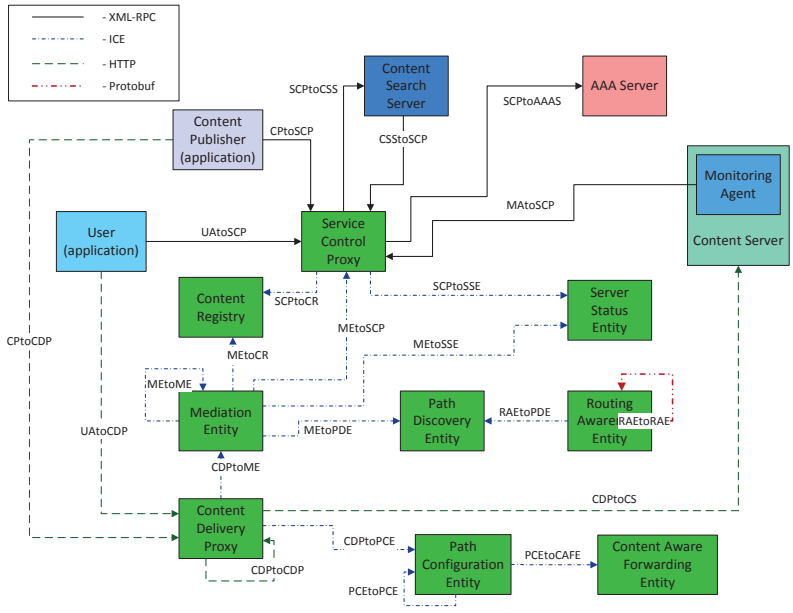
Następnie moduł CDP przesyła żądanie pobrania treści do modułu CDP dołączonego do sieci dostępowej, w której znajduje się wybrany serwer CS (krok 11). Moduł CDP serwera uruchamia proces konfiguracji ścieżki pomiędzy CDP użytkownika i serwera wysyłając żądanie konfiguracyjne do modułu PCE (ang. *Path Configuration Entity* - krok 12). Zadaniem modułu PCE jest wytypowanie listy kluczy (krok 13) wskazujących, którymi węzłami przekazu danych sieci PI CAN, nazywanymi CAFE (ang. *Content Aware Forwarding Entity*), przesyłane będą ramki zawierające pobieraną treść. Następnie PCE przeprowadza konfigurację brzegowego węzła CAFE, do którego dołączony jest serwer treści. Węzeł ten ma za zadanie przechwycenie ramek wysyłanych przez serwer treści do użytkownika i dodanie do nich nagłówka z listą kluczy otrzymaną od PCE. W ten sposób każda ramka wysyłana przez serwer otrzymuje na wejściu do sieci PI CAN nagłówek z informacją definiującą ścieżkę przekazu pakietów poprzez węzły CAFE w sieci PI CAN (krok 14 i 15).

Po ukończeniu procesu konfiguracji brzegowego węzła CAFE (krok 16), żądanie pobrania treści przesyłane jest do wybranego serwera CS (krok 17). Po otrzymaniu żądania serwer rozpoczyna transmisję treści do użytkownika (kroki 18, 19 i 20).

## 4 Realizacja płaszczyzny sterowania

Na rys. 3 zaprezentowano moduły funkcjonalne wchodzące w skład architektury sieci PI CAN. Możemy pogrupować je na dwa obszary: elementy zewnętrzne współpracujące z siecią PI CAN, oraz elementy wewnętrzne sieci PI CAN. Do grupy elementów zewnętrznych zaliczamy, oprócz serwerów przedstawionych w paragrafie 2 (tj. wyszukiwania CSS, uwierzytelniania i autoryzacji AAAS oraz przechowującego treści CS), aplikację użytkownika UA (ang. *User Application*) wykorzystywaną do wyszukiwania i pobierania treści oraz aplikację dostawcy treści CP (ang. *Content Publisher*) używaną do publikacji, modyfikacji i usuwania treści z sieci PI CAN.

Grupa elementów wewnętrznych obejmuje następujące moduły: 1) SCP, pośredniczący w wymianie wiadomości pomiędzy elementami zewnętrznymi i siecią PI CAN w procesach zarządzania treścią i wyszukiwania treści, 2) CDP, pośredniczący w procesie pobierania treści pomiędzy siecią PI CAN a aplikacją użytkownika/serwerem CS, 3) CR, przechowujący rekordy opublikowanej treści, 4) ME, odpowiedzialny za wybór optymalnego serwera i ścieżki w sieci dla realizacji transmisji treści, 5) PCE, odpowiadający za konfigurację ścieżek w sieci, 6) SSE



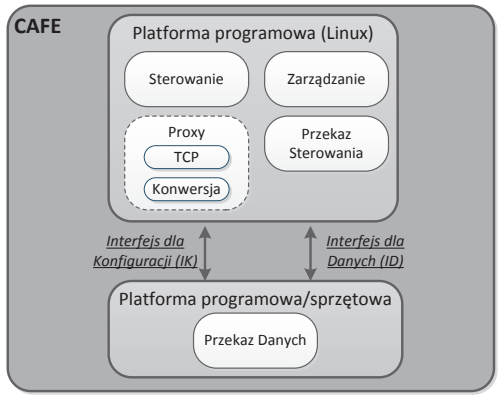

**Rysunek 3.** Moduły funkcjonalne systemu PI CAN

monitorujący stan obciążenia serwerów treści, 7) PDE wyszukujący dostępne w sieci ścieżki domenowe pomiędzy siecią dostępową użytkownika a serwerem treści, 8) RAE (ang. *Routing Awareness Entity*), którego zadaniem jest pozyskiwanie informacji o aktualnie zestawionych w sieci ścieżkach rutingu.

Przedstawione powyżej moduły wewnętrzne sieci PI CAN stworzone zostały jako maszyny wirtualne (pod kontrolą systemu operacyjnego Linuks) uruchamiane na węzłach Systemu IIP z zainstalowanym monitorem maszyn wirtualnych Xen. Moduły te zostały zaimplementowane przy wykorzystaniu następujących technologii:

- język C++ z użyciem biblioteki Boost [13] (SCP, CR, RAE, SSE);
- język Java z użyciem systemu zarządzania relacyjnymi bazami danych Apache Derby [14] (PDE, PCE);
- język Python (ME);
- język Python z użyciem platformy Tornado [15] (CDP).

Do realizacji interfejsów płaszczyzny sterowania dla elementów zewnętrznych (UAtoSCP, CPtoSCP, SCPtoCSS, CSStoSCP, SCPtoAAAS, MAtoSCP) wykorzystana została technologia XML-RPC [16], przez wzgląd na jej prostotę i duże rozpowszechnienie. Biblioteki XML-RPC dostępne są dla wielu języków programowania, m.in. C, C++, Java, Python, Perl, JavaScript, Ruby. Dzięki




**Rysunek 4.** Model funkcjonalny węzła CAFE

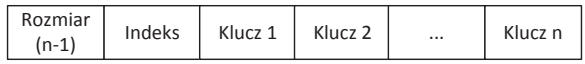
temu technologia zastosowana do implementacji interfejsu płaszczyzny sterowania pozostawia twórcom elementów zewnętrznych szerokie możliwości wyboru platformy dla tworzonych przez nich aplikacji.

Z kolei implementacja interfejsów płaszczyzny sterowania dla elementów wewnętrznych (SCPtoCR, SCPtoSSE, CDPtoME, MEtoME, MEtoCR, MEtoSCP, MEtoPDE, MEtoSSE, RAEtoPDE, CDPtoPCE, PCEtoPCE, CAFEtoPCE) została wykonana przy zastosowaniu platformy Zeroc Ice [17]. Użycie platformy Ice skutkuje możliwością implementacji elementów wewnętrznych jedynie w sześciu wybranych językach programowania (w tym C++, Java i Python), jednakże platforma ta cechuje się dużo większą wydajnością i efektywnością (większa szybkość przekazu wiadomości, mniejsze zapotrzebowanie na zasoby) w porównaniu do innych technologii, jak XML-RPC, SOAP czy CORBA [18].

Wybór technologii dla implementacji interfejsów płaszczyzny danych (UAtoCDP, CPtoCDP, CDPtoCDP, CDPtoCS) uzależniony jest od protokołu zastosowanego do transmisji treści z serwera CS. Wykonana implementacja systemu PI CAN pozwala użytkownikowi pobierać treści wykorzystując protokół HTTP [19]. W celu użycia innych protokołów strumieniujących, np. RTP [20], konieczna jest implementacja obsługi tych protokołów w module CDP oraz serwerze treści CS.

## 5 Realizacja płaszczyzny przekazu danych

Płaszczyzna danych sieci PI CAN składa się z węzłów CAFE, które pełnią funkcję analogiczną do ruterów IP. Węzły te są odpowiedzialne za przekaz treści z serwerów treści do konsumentów treści wzdłuż ustalonych ścieżek, oraz przekaz wiadomości sterujących i zarządzających.

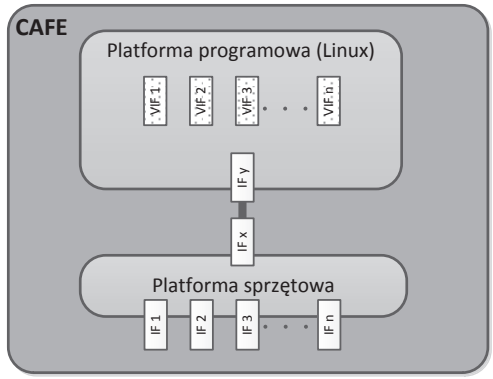



**Rysunek 5.** Nagłówek PI CAN

Rys. 4 przedstawia ogólny, niezależny od platformy sprzętowej, model funkcjonalny węzła. Model ten wyróżnia następujące bloki funkcjonalne: *Sterowanie*, *Zarządzanie*, *Proxy*, *Przekaz Sterowania* (bloki zrealizowane programowo), oraz *Przekaz Danych* (blok zrealizowany programowo oraz sprzętowo). Moduł *Przekaz Danych* odpowiedzialny jest za przekaz pakietów danych sieci PI CAN przenoszących treść. Przekaz pakietów danych sieci PI CAN odbywa się na podstawie wartości pól nagłówka PI CAN oraz tablicy przekazu danych. Nagłówek PI CAN (rys. 5) składa się z listy kluczy oraz pól *Rozmiar* i *Indeks*. Pole *Rozmiar* definiuje długość listy kluczy, natomiast pole *Indeks* wskazuje na bieżący klucz. Każdy węzeł CAFE na podstawie bieżącego klucza w nagłówku komutuje pakiet na odpowiedni interfejs wyjściowy, po czym aktualizuje wartość pola *Indeks*.

Główną funkcją modułu *Przekaz Sterowania* jest przekaz datagramów IPv6 przenoszących wiadomości sterujące oraz zarządzające węzłami sieci PI CAN. Moduł *Proxy* znajduje się wyłącznie w brzegowych węzłach CAFE i składa się z dwóch elementów: *Konwersji* oraz *TCP*. Element *Konwersja* służy do konwersji pomiędzy formatem danych używanym po stronie konsumenta/serwera treści (datagramy IPv6) oraz formatem używanym w sieci szkieletowej (pakiety PI CAN). Natomiast element *TCP* zajmuje się utrzymaniem połączeń TCP służących do przekazu danych. Przy pobieraniu treści wykorzystywane są trzy połącznia TCP: 1) pomiędzy konsumentem treści a węzłem brzegowym CAFE po stronie konsumenta treści, 2) pomiędzy węzłami brzegowymi CAFE, oraz 3) pomiędzy serwerem treści, a brzegowym węzłem CAFE po stronie serwera. Moduł *Sterowanie* pełni funkcję procesu rutingu (dla IPv6), natomiast moduł *Zarządzanie* jest odpowiedzialny za zdalną konfigurację pozostałych modułów (np. konfigurację tablic przekazu danych w module *Przekaz Danych*).

Węzeł CAFE został zrealizowany programowo na platformie Xen (zobacz pkt. 5.1) oraz w oparciu o platformy sprzętowe EZappliance (zobacz pkt. 5.2) oraz NetFPGA (zobacz pkt. 5.3). Platformy sprzętowe zostały wykorzystane w celu zagwarantowania wydajnego przetwarzania pakietów. Z uwagi na fakt, że wydajność modułu *Przekaz Danych* ma kluczowe znaczenie dla sprawności całego węzła CAFE (moduł ten jest odpowiedzialny za przekaz większości ruchu), został on zaimplementowany sprzętowo. Moduły *Zarządzanie*, *Sterowanie* oraz *Przekaz Sterowania* przetwarzają tylko niewielki ułamek ruchu, dlatego ich programowa realizacja jest wystarczająca dla wydajnego działania węzła CAFE. Moduł *Proxy* został również zrealizowany wyłącznie programowo gdyż jest on wykorzystywany jedynie w brzegowych węzłach CAFE dla obsługi ruchu wchodzącego/wychodzącego z sieci szkieletowej.



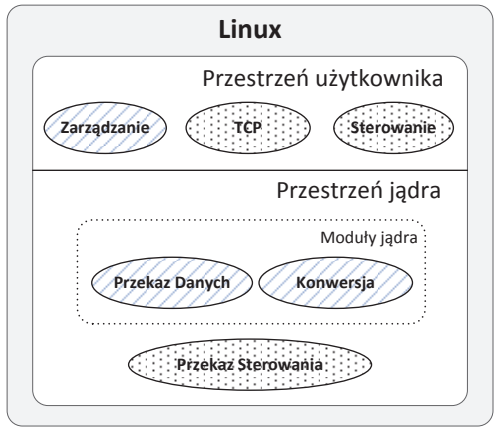

**Rysunek 6.** Model fizyczny węzła CAFE opartego na platformę sprzętową

Platformy sprzętowa i programowa są połączone za pomocą dwóch interfejsów: *Interfejsu dla Konfiguracji*(IK) oraz *Interfejsu dla Danych* (ID). Interfejs IK służy do konfiguracji tablic przekazu danych oraz do konfiguracji mechanizmów zarządzania ruchem. Natomiast interfejs ID jest wykorzystywany do przekazu dwóch rodzajów pakietów: 1) przenoszących wiadomości sterujące i zarządzające, oraz 2) przenoszących treść (tylko w brzegowych węzłach CAFE).

Na rys. 6 został przedstawiony model fizyczny węzła CAFE opartego na platformie sprzętowej. Fizyczne interfejsy danych platformy sprzętowej (*IF 1*, *IF 2*, *IF 3*, ..., *IF n*) są odzwierciedlone na platformie programowej w postaci interfejsów wirtualnych VLAN IEEE 802.1Q (*VIF 1*, *VIF 2*, *VIF 3*, ..., *VIF n*). Pakiety PI CAN przesyłane pomiędzy platformami poprzez logiczny interfejs ID są oznaczane odpowiednimi znacznikami VLAN w zależności od interfejsu fizycznego na którym zostały odebrane lub którym mają zostać wysłane. Odwzorowanie pomiędzy znacznikami VLAN i interfejsami fizycznymi jest konfigurowane w module *Przekaz Danych* za pomocą logicznego interfejsu IK. Dzięki wykorzystaniu standardowych interfejsów wirtualnych VLAN w platformie programowalnej wszystkie moduły programowe współpracują przezroczyście z platformą sprzętową.

### 5.1 Realizacja programowa - platforma Xen

Podstawową realizacją węzła CAFE jest realizacja programowa w postaci maszyny wirtualnej (dla monitora maszyn wirtualnych Xen) opartej o system operacyjny Linux. Schemat realizacji programowej został przedstawiony na rys. 7 - wypełnieniem z liniami zostały oznaczone moduły zaimplementowane od postaw, natomiast wypełnieniem z kropkami zostały zaznaczone moduły zrealizowane z wykorzystaniem istniejącego oprogramowania.



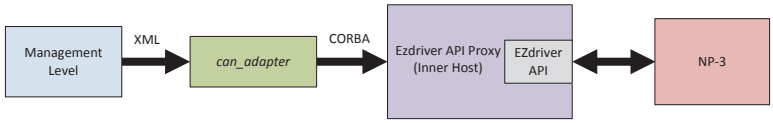

**Rysunek 7.** Programowa realizacja węzła CAFE

Dla realizacji programowej kluczową rolę pełni zapewnienie możliwie wysokiej efektywności przekazu danych. W celu jej osiągnięcia elementy *Przekaz Danych* oraz *Konwersja* zostały zaimplementowane jako ładowalne moduły jądra systemu Linux. Moduł *Zarządzanie* został zrealizowany jako program w przestrzeni użytkownika w języku Python 2.7. Komunikacja pomiędzy programem *Sterowanie* a modułami jądra (*Przekaz Danych*, *Konwersja*) odbywa się za pomocą programów zaimplementowanych w języku C opartych na bibliotece *libnl*. Komunikacja płaszczyzny sterowania z węzłami CAFE (z modułem *Sterownie*) oparta jest na interfejsie zrealizowanym z wykorzystaniem technologii *Protocol Buffers* [21].

Moduł *Przekaz Sterowania* wykorzystuje standardową logikę IPv6 w systemie Linux, natomiast moduł *Sterowanie* został zrealizowany w oparciu o router Quagga [22]. Moduł *TCP* oparty jest na serwerze Tornado [15].

### 5.2 Realizacja w oparciu o platformy sprzętowe - EZappliance

Urządzenie EZappliance [23] to programowalna platforma sprzętowa dedykowana do szybkiego testowania i wdrażania nowych rozwiązań sieciowych. EZappliance bazuje na procesorze sieciowym EZchip NP-3 [24] o wydajności przetwarzania ramek wynoszącej 30 Gbps. Procesor ten składa się z potoku procesorów zorientowanych zadaniowo TOP (ang. *Task-Optimized Processor*): TOPparse, TOPsearchI, TOPresolve, TOPsearchII, TOPmodify, oraz z dwóch modułów służących do zarządzania ruchem TM (ang. *Traffic Manager*). Procesory TOP odpowiedzialne są za: wyodrębnienie z nagłówków pakietów pól protokołów (adresy, porty, itp.), wyznaczanie interfejsów wyjściowych (na podstawie tablic tra-




**Rysunek 8.** Rola oprogramowania *can_adapter* w PI CAN

sowania) oraz modyfikację pakietów. Moduły TM umożliwiają elastyczne kształtowanie ruchu wychodzącego z procesora NP-3.

Procesor sieciowy NP-3 w EZappliance jest zarządzany przez dedykowany system komputerowy (wbudowany w EZappliance), nazwany dalej IH (*Inner Host*), oparty o procesor Power PC i system operacyjny Linux. System ten dostarcza interfejs EZdriver API umożliwiający konfigurację procesora NP-3, definiowanie i zarządzanie wewnętrznymi strukturami danych, ładowanie i uruchamianie programów dla NP-3 czy obsługę przerwań sprzętowych.Realizacja węzła wirtualnego sieci PI CAN jest zgodna z ogólnym rozwiązaniem przyjętym dla urządzenia EZappliance w Systemie IIP [25]. Oprogramowanie węzła CAFE dla urządzenia EZappliance składa się dwóch podstawowych części: 1) oprogramowania *can_forwader* dla procesorów TOP, które realizuje funkcję *Przekaz Danych*, oraz 2) oprogramowania *can_adapter* dla systemu IH, które realizuje funkcję *Zarządzania*.

Przy realizacji węzła CAFE na urządzeniu EZappliance wykorzystywana jest również dodatkowa platforma programowa oparta na systemie z rodziny Linux, na której umieszczone są następujące moduły oprogramowania (patrz rys. 8): *Sterowanie*, *Proxy* (w przypadku brzegowych węzłów CAFE), *Przekaz Sterowania*, *Zarządzanie*.

Oprogramowanie *can_forwarder* operuje na portach wirtualnych zapewnionych przez warstwę wirtualizacji Systemu IIP [11]. Zostały rozróżnione następujące rodzaje portów wirtualnych: *access*, *core* oraz *control*. Porty typu *access* służą do podłączenia sieci dostępowych (w tym serwerów treści); za pomocą portów typu *core* łączone są sąsiadujące węzły CAFE, natomiast porty typu *control* są wykorzystywane do podłączenia platformy sprzętowej z platformą programową. Porty typu *access* są używane tylko w brzegowych węzłach CAFE, ramki odbierane na tych portach są przekazywane do platformy programowej (do modułu *Proxy*) poprzez porty typu *contorl* (z wykorzystaniem logicznego interfejsu ID). Każdemu portowi typu *access* w module *can_forwarder* odpowiada port VLAN na platformie programowej (moduł *Przekaz Danych* odpowiednio znakuje pakiety znacznikami VLAN). Dzięki wykorzystaniu portów VLAN moduły platformy programowej są takie same jak w realizacji z wykorzystaniem platformy Xen (zobacz pkt. 5.1).

Oprogramowanie *can_forwarder* wykorzystuje dwie tablice przekazu danych: *can_tag_forward* i *can_port_forward*, które zostały zrealizowane jako tablice ha-

szujące (ang. *hash table*). Tablica *can_tag_forward* służy do przełączania ramek posiadających nagłówek sieci PI CAN. Klucz wyszukiwania w tej tablicy składa się z: a) identyfikatora wirtualnego portu, którym przyszła ramka, oraz b) klucza przełączania z nagłówka PI CAN. Rezultat wyszukiwania zawiera natomiast: a) identyfikator portu wirtualnego, którym ma zostać wysłana ramka, b) identyfikator kolejki wyjściowej (kolejki są wykorzystywane przez mechanizmy zarządzania ruchem), oraz c) wartość opcjonalnego znacznika VLAN (nagłówek 802.1Q jest dodawany do ramki wyłącznie gdy ma ona zostać wysłana do platformy programowej).

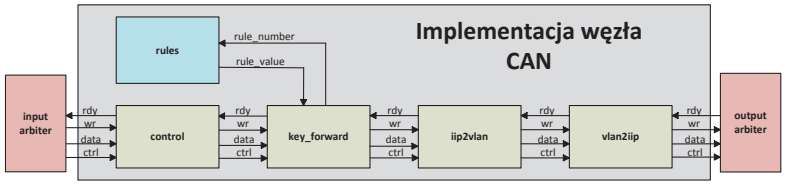
Tablica *can_port_forward* jest wykorzystywana do przełączania ramek, które nie posiadają nagłówka PI CAN (np. ramki przychodzące z sieci dostępowej, ramki zawierające wiadomości sterujące). Klucz wyszukiwania w tablicy *can_port_forward* składa się z: a) identyfikatora wirtualnego port, którym przyszła ramka, oraz b) wartości znacznika VLAN w ramce. Format rezultatu w tej tablicy jest identyczny z formatem rezultatu w tablicy *can_tag_forward*.

Oprogramowanie *can_forwarder* wykorzystuje potok czterech procesorów TOP. Procesor TOPparse służy do wyodrębnienia z pakietów/ramek PI CAN pól nagłówków. W szczególności dla ramek odebranych na portach typu *control* odczytywane są znaczniki VLAN, natomiast dla ramek danych PI CAN odebranych na portach typu *control* odczytywane są bieżące klucze przekazu danych. Na podstawie wyodrębnionych pól nagłówków tworzone są klucze wyszukiwania w tablicach przekazu danych. Procesor TOPsearchI służy do wyszukania rezultatu w strukturach przekazu danych, który określa m.in. wyjściowy port wirtualny.

Podczas przetwarzania ramki w procesorze TOPresolve zostaje podjęta decyzja w jaki sposób należy zmodyfikować ramkę. Możliwe są następujące modyfikacje ramek: dodanie nagłówka 802.1Q, usuńcie nagłówka 8021Q, inkrementacja klucza przekazu danych w nagłówku PI CAN. Ostatni etap przetwarzania ramek to ich modyfikacja oraz wysyłka poprzez odpowiednie porty wirtualne, za co odpowiedzialny jest procesor TOPmodify.

Rolę styku pomiędzy oprogramowaniem *can_forwarder* a platformą programową pełni oprogramowanie *can_adapter*. Zadaniem modułu *can_adapter* jest przetwarzanie pliku XML zawierającego zadaną konfigurację tablic przekazu danych oraz mechanizmów zarządzania ruchem, a następnie przypisanie tych wartości strukturom NP-3 przy użyciu funkcji dostarczanych przez EZdriver API dostępnych na IH. Rolę oprogramowania *can_adapter* obrazuje rys. 6. Ze względu na to, że IH dysponuje niewielką ilością zasobów pamięciowych oraz mocą obliczeniową, aby spełnić wymagania stawiane przez PI CAN (oraz system IIP) konieczne jest użycie dodatkowego, zewnętrznego systemu komputerowego, nazywanego dalej OH (*Outer Host*), pozbawionego ograniczeń IH.

W celu umożliwienia OH realizacji funkcjonalności płaszczyzny sterowania należy zapewnić dostęp do rejestrów i pamięci procesora NP-3. Jednak taka funkcjonalność jest dostępna jedynie na IH poprzez lokalne wywołania metod EZdriver API. Dlatego w niniejszej realizacji węzła sieci PI CAN system IH pełni jedynie rolę pośrednika pomiędzy procesorem NP-3 a OH, udostępniając




**Rysunek 9.** Schemat blokowy implementacji modułu *can_forwarder*

funkcje EZdirver API jako zdalnie wywoływane procedury zaimplementowane w technologii CORBA - jako EZdriver API Proxy.

Oprogramowanie *can_adapter* zostało zrealizowane jako proces działający w tle (ang. *daemon*), który nieustannie śledzi zmiany w pliku XML, dokonuje ich walidacji i propaguje je (inkrementacyjnie) do procesora NP-3.

### 5.3 Realizacja w oparciu o platformy sprzętowe - NetFPGA

Karta NetFPGA wyposażona w układ programowalny oraz cztery porty Ethernet w standardzie 1GE posłużyła do sprzętowej implementacji elementów węzła CAFE. Oprogramowanie węzła CAFE dla karty NetFPGA składa się z dwóch podstawowych części: 1) oprogramowania *can_forwader*, które realizuje funkcję *Przekaz Danych*, oraz 2) oprogramowania *can_adapter* które realizuje funkcję *Zarządzania*. Na karcie NetFPGA został zaimplementowany moduł *can_forwarder*, pozostałe moduły (tj. *Sterowanie*, *Proxy*, *Przekaz Sterowania*, *Zarządzanie*) znajdują się na komputerze z system Linux (platformie programowa), w którym zainstalowana jest karta NetFPGA. Moduł *can_forwarder* został zrealizowany w ramach architektury pokazanej na rys. 9. Moduł ten składa się z następujących bloków: *control*, *key_forward*, *iip2vlan*, *vlan2iip*, *rules*.

Moduł *can_forwarder* został wpasowany w ścieżkę przetwarzania danych *User Data Path*, zaproponowaną przez producenta karty (Stanford University). Wykorzystanie referencyjnego projektu pozwoliło skupić się na realizacji zadania. Uniknięto tym samym zbędnej implementacji mechanizmów zarządzających komunikacją z kontrolerami MAC, złączem PCI oraz konfiguracją bloków funkcjonalnych wchodzących w skład pracującego węzła. Warto podkreślić, że *User Data Path* wspiera model sekwencyjnego przetwarzania danych. Tym samym możliwy stał się podział na cztery niezależne od siebie bloki sprzętowe (rys. 9). Należy zaznaczyć, że istnieją wyjątki wynikające z równań logicznych, które przewidują równoczesną aktywność dwóch elementów kaskady. Każdy element składowy obsługuje tryb bypass - co oznacza, że może on być w pełni transparentny i przekazywać dane bez wprowadzania jakichkolwiek zmian w treści pakietu. Uzyskano w ten sposób optymalną postać RTL implementowanego węzła, co pozwoliło utrzymać częstotliwość taktowania na poziomie 125 MHz. Był

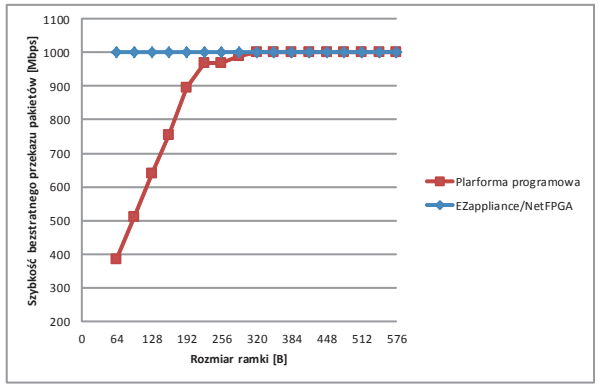
to warunek konieczny, aby platforma NetFPGA mogła pracować z pełną wydajnością.

Skrajny element łańcucha - tj. blok, opatrzony na rys. 9 oznaczeniem *control*, pełni rolę dyspozytora, który na podstawie wstępnej analizy przekazywanego strumienia, decyduje o aktywności pozostałych bloków logicznych, formujących ścieżkę przetwarzania danych. Przedstawiona koncepcja wiąże się zarówno z rozbudową założeń, przyjętych przez Stanford University w zakresie interpretacji sygnałów magistrali sterującej, jak również ze swoistą dwustanowością bloków logicznych. Zmiana sygnalizacji ma charakter lokalny i obejmuje modyfikację trzech najmniej znaczących elementów (bitów) słowa sygnalizującego początek ramki. Pojedynczy sygnał tak uformowanego (trzybitowego) słowa, jest statycznie związany z konkretnym blokiem, który na podstawie przypisanego mu sygnału podejmuje decyzję o pasywnym przekazaniu strumień do następnika, lub przetwarza strumień według właściwości, z uwzględnieniem przywrócenia prawidłowego stanu linii słowa kontrolnego na wyjściu.

Dwa komplementarne bloki logiczne *vlan2iip* oraz *iip2vlan*, nadają prawidłową formę ramce, przed przekroczeniem przez nią granicy styku z programową częścią węzła CAFE. Produktem wyjściowym jest - w zależności od kierunku migracji - pakiet połączony z nagłówkiem transportowym niższych warstw transmisyjnych, bądź pozbawiony odpowiedniego nagłówka, przed przekazaniem go do programowej części węzła PI CAN. Modyfikacji ulega zarówno długość, jak i przeznaczenie pakietu. Praca bloków logicznych opiera się na przetwarzaniu potokowym, gwarantującym ściśle określone opóźnienie propagacyjne. Z przyczyn naturalnych, przekształcenie związane z produkcją strumienia przewyższającego długością strumień wejściowy (tj. powstały w wyniku transmisji danych w relacji CAFE $\rightarrow$ otoczenie zewnętrzne), wymusza blokadę przyjmowania nowej ramki, na czas związany z transmisją treści dodanej, który obejmuje maksymalnie 2 cykle zegarowe (jest uzależniony od wyrównania pakietu w stosunku do 64-bitowego słowa, formującego właściwą treść strumienia). Z punktu widzenia elektroniki cyfrowej, zadania realizowane przez bloki mają charakter złożony i nie sprowadzają się wyłącznie do prostego skracania, czy konkatenacji ciągów. Należy wziąć pod uwagę zmiany wprowadzane w obszarze VLAN (kodowanie źródła/wybór interfejsu), oraz dekonstrukcję strumienia w sferze abstrakcji NetFPGA (manipulowanie długością strumienia wiąże się z ustanowieniem nowej, zmienionej treści w kanale CTRL). Opisywane funkcje można określić mianem sprzętowej akceleracji w zakresie czynności enkapsulacyjno/dekapsulacyjnych, prowadzonych przez węzeł.

Moduł *key_forward* przetwarza ramki przesyłanych pomiędzy węzłami sieci PI CAN, tzn. takie, które nie przekraczają granicy styku z programową częścią węzła CAFE. Ramki te są przenoszone pomiędzy pośrednimi węzłami. Przetwarzanie obejmuje odczyt bieżącego indeksu klucza, pobranie reguły z tablicy reguł, zwiększenie indeksu klucza o jeden i skierowanie pakietu do interfejsu zgodnego z odczytaną wartością reguły.

Indeks klucza niezbędny do określenia wyjściowego interfejsu znajduje się wewnątrz ramki, a jego pozycja jest zmienna, więc konieczne jest buforowanie




**Rysunek 10.** Szybkość bezstratnego przekazu pakietów danych PI-CAN

przychodzącego pakietu w pamięci wewnętrznej układu programowalnego. Odebranie fragmentu ramki zawierającego indeks klucza umożliwia odczyt reguły, określenie interfejsu i przesłanie zbuforowanej ramki do właściwego węzła.

Tablica reguł jest oparta o dwuportową pamięcią RAM wydzieloną z zasobów układu FPGA. Zapisywanie reguł przetwarzania ramek PI CAN odbywa się z poziomu systemu operacyjnego komputera. Służy do tego moduł *can_adapter*, zdolny do operacji na rejestrach przestrzeni adresowej platformy NetFPGA. Odwołania do poszczególnych rejestrów są przechwytywane przez rdzeń *key_forward* i tam zapisywane bądź odczytywane są dane w zależności od rodzaju operacji.

### 5.4 Wydajność płaszczyzny przekazu danych

Wydajność różnych implementacji węzła CAFE została oceniona poprzez pomiar maksymalnej szybkości bezstratnego przekazu pakietów danych PI CAN. Generator ruchu (Spirent TestCenter [26]) wysyłał strumień pakietów danych PI CAN do węzła CAFE z szybkością łącza równą 1 Gbps. Następnie strumień pakietów był komutowany przez CAFE na interfejs wyjściowy na podstawie nagłówka PI CAN (wartości bieżącego klucza w nagłówku PI CAN). Interfejs wyjściowy CAFE był podłączony do monitora ruchu (Spirent Test Center), który analizował odebrany strumień pakietów. Rys. 10 przedstawia charakterystyki maksymalnej szybkości bezstratnego przekazu pakietów danych PI CAN dla różnych implementacji. Otrzymane wyniki pokazują, że wydajność platformy programowej zależy wprost proporcjonalnie od wielkości pakietów. Badana platforma programowa bezstratnie przekazywała pakiety większe niż 384B (z szybkością łącza). Natomiast, platformy sprzętowe są zdolne do bezstratnego przekazu pakietów danych PI CAN (z szybkością łącza) niezależnie od ich rozmiaru.

## 6 Podsumowanie

Artykuł prezentuje realizację sieci świadomej treści PI CAN opracowanej w ramach projektu Inżynieria Internetu Przyszłości. Sieć PI CAN jest zaimplementowana w oparciu o wirtualną infrastrukturę udostępnianą przez System IIP, tworzoną za pomocą platformy programowej (Xen) oraz platform sprzętowych (EZappliance i NetFPGA). Artykuł przedstawia płaszczyznę sterowania systemu PI CAN, która jest wyspecyfikowana poprzez opis interfejsów pomiędzy modułami wewnętrznymi, jak również interfejsów do elementów zewnętrznych współpracujących z siecią PI CAN. Ponadto pokazano implementację płaszczyzny przekazu danych, która została zrealizowana na trzech ww. platformach: programowej Xen oraz sprzętowych EZappliance i NetFPGA. Zaprezentowane testy demonstrują, iż opracowane węzły przekazu danych sieci PI CAN dla interfejsów 1 Gbps są zdolne przekazywać pakiety z pełną szybkością łącza.

## Literatura


1. V. Jacobson, et al., "Networking Named Content", CoNEXT 2009, Rome, Italy, December 2009
2. H. Yuan, T. Song and P. Crowley, "Scalable NDN Forwarding: Concepts, Issues and Principles", In Proceedings of the 21th International Conference on Computer Communications and Networks (ICCCN), Munich, Germany, August 2012
3. T. Koponen, et al., "A data-oriented (and beyond) network architecture". In Proc. ACM SIGCOMM '07, pages 181-192, New York, NY, USA, 2007. ACM Press.
4. D. Kutscher, S. Farrell, and E. Davies. The NetInf Protocol. Internet-Draft draft-kutscher-icnrg-netinf-proto, Internet Engineering Task Force, October 2012. Work in progress.
5. Stephen Farrell, Dirk Kutscher, Christian Dannewitz, Börje Ohlman, and Phillip Hallam-Baker. Naming Things With Hashes. Internet Draft draft-farrell-decade-ni, Work in progress, August 2012
6. M. D. Ambrosio, et al., "Second NetInf architecture description", Deliverable 6.2 of project 4WARD, FP7-ICT-2007-1-216041-4WARD/D-6.2, January 2010
7. "Final NetInf Architecture", Deliverable D.B.3 of project SAIL - Scalable and Adaptable Internet Solutions, FP7-ICT-2009-5-257448/D.B.3, January 2013
8. Projekt POIG.01.01.02-00-045/09-00 Inżynieria Internetu Przyszłości, `https://www.iip.net.pl`
9. W. Burakowski, H. Tarasiuk, A. Bęben, "Architektura Systemu IIP", Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011)
10. A. Bęben i inni, "Architektura sieci świadomych treści w systemie IIP", Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011)
11. A. Beben, et al., "Content Aware Network based on Virtual Infrastructure", Proc. of 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2012), Kyoto, Japan, August 2012
12. B. Belter i inni, "Warstwa sieci w sieciach świadomych treści", Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011)
13. Boost C++ Libraries. Strona www projektu: `http://www.boost.org`
14. Apache Derby database. Strona www projektu: `http://db.apache.org/derby/`

15. Tornado web server. Strona internetowa projektu: http://www.tornadoweb.org
16. Specyfikacja XML-RPC, dostępna on-line: http://xmlrpc.scripting.com/spec (dostęp: marzec 2013)
17. Internet Communications Engine (Ice) firmy Zeroc, strona www: http://www.zeroc.com
18. M. Henning i M. Spruiell, "Choosing Middleware: Why Performance and Scalability do (and do not) Matter", raport techniczny firmy Zeroc, dostępny online (dostęp: marzec 2013): http://www.zeroc.com/articles/IcePerformanceWhitePaper.pdf
19. R. Fielding et al., "Hypertext Transfer Protocol – HTTP/1.1", IETF RFC 2616, June 1999
20. H. Schulzrinne et al., "RTP: A Transport Protocol for Real-Time Applications", IETF RFC 3550, July 2003
21. Protocol Buffers. Strona www projektu: https://developers.google.com/protocol-buffers/
22. Strona internetowa projektu Quagga: http://www.nongnu.org/quagga
23. EZchip, "EZappliance product brief" dostępny na stronie (dostęp: marzec 2013): http://www.ezchip.com/Images/pdf/NP-3_EZappliance_Brief_online.pdf
24. EZchip, "NP-3 product brief" dostępny na stronie (dostęp: marzec 2013): http://www.ezchip.com/Images/pdf/NP-3_Short_Brief_online.pdf
25. P. Wiśniewski, D. Parniewicz, K. Dombek, A. Juszczyk, I. Olszewski, "Implementacja wirtualizacji dla Systemu IIP na platformie EZappliance", Przeglad Telekomunikacyjny i Wiadomosci Telekomunikacyjne, SIGMA NOT, 2012, nr 8-9 (t. CD-ROM), s. 1387-1398
26. Spirent TestCenter. Strona www produktu: http://www.spirent.com/Ethernet_Testing/Software/TestCenter

# Routing i wymiarowanie w sieci świadomej treści

Paweł Białoń[1], Janusz Granat[1], Paweł Olender[1], Andrzej Beben[2], Piotr Krawiec[1], Piotr Wiśniewski[2], Bartosz Belter[3], Jakub Gutkowski[3], Łukasz Łopatowski[3]

[1] Instytut Łączności - Państwowy Instytut Badawczy
[2] Politechnika Warszawska, Instytut Telekomunikacji
[3] Poznańskie Centrum Superkomputerowo-Sieciowe

**Streszczenie** Artykuł przedstawia hierarchiczny mechanizm routingu w sieciach świadomych treści (CAN) stworzony w ramach projektu Inżynieria Internetu Przyszłości W prezentowanym podejściu problem wyznaczania tras w ramach domeny CAN sformułowany jest w postaci zadania optymalizacyjnego, w którym uwzględniane są wymagania dotyczące przekazu treści, zapotrzebowania użytkowników oraz dostępność zasobów w ramach domeny. Z drugiej strony, trasy między domenami są wyznaczane w sposób rozproszony przez wielokryterialny i wielościeżkowy protokół routingu, w którym informacje o dostępności ścieżek i ich parametrach są wymieniane pomiędzy sąsiednimi domenami. Rzeczywiste ścieżki przesyłu treści są wybierane dla każdego żądania za pomocą dedykowanego modułu konfiguracji ścieżek, który wykorzystuje w tym celu informacje udostępnione przez moduły routingu wewnątrz i między domenowego. Zaproponowane rozwiązanie zostało zilustrowane przykładowymi wynikami.

## 1 Wprowadzenie

U podstaw koncepcji sieci świadomych treści (ang. *Content Aware Network*, CAN) leży ich ukierunkowanie i specjalizacja w przesyle treści multimedialnych. Sieć CAN została zaprojektowana w celu wspierania dostępu i dostarczania treści multimedialnych oraz wyeliminowania znanych ograniczeń dzisiejszego Internetu [1] związanych z: (1) brakiem globalnego schematu nazewnictwa treści, (2) niewydajnym dostarczaniem treści ze względu na brak świadomości treści, (3) brakiem połączeń typu jeden-do-najbliższego (anycast) i jeden-do-wielu (multicast) oraz (4) problemem niedziałających odnośników spowodowanym uzależnieniem od lokalizacji. W ostatnim czasie badaniom w zakresie sieci CAN poświęcono wiele uwagi w ramach różnych inicjatyw, np. DONA [2], CCN [3], 4WARD/SAIL [4], PSIRP/PURSUIT [5] oraz COMET [6]. Jednym z kluczowych wyzwań w obszarze badań sieci CAN jest zaprojektowanie routingu i inżynierii ruchu świadomych treści z uwzględnieniem wymagań na przekaz treści, lokalizacji kopii, popularności poszczególnych treści oraz zapotrzebowania użytkowników i zasobów infrastruktury sieciowej. Dotychczasowe rozwiązania powielają podejścia oparte na routowaniu na podstawie nazwy (ang. "*route by name*") [3], [7] lub bazujące

na serwerach nazw (DNS) [4], [6]. Routowanie na podstawie nazwy zakłada kierowanie żądań do serwerów w oparciu o identyfikator treści, np. nazwę. Z kolei podejście z serwerami DNS zakłada pobieranie treści dwuetapowo. W pierwszym etapie żądanie jest przekształcane na adres serwera treści. W drugim etapie klient pobiera treść wprost z serwera wybranego w etapie pierwszym.

W artykule opisujemy kluczowe elementy algorytmów inżynierii ruchu i protokołów zaprojektowanych dla sieci CAN w ramach projektu Inżynieria Internetu Przyszłości [8]. Więcej informacji na temat projektu i rozwijanej sieci CAN można znaleźć odpowiednio w [8] oraz [9], [10].

W proponowanym rozwiązaniu, bazującym na koncepcji routingu źródłowego, można wyróżnić złożone specjalizowane operacje przebiegające w wielu warstwach i wielu etapach. Po pierwsze, w prezentowanym routingu i inżynierii ruchu należy wydzielić dwa rodzaje zadań. Jedna grupa to zadania wymagające zazwyczaj złożonych i czasochłonnych obliczeń, które w związku z tym wykonywane są zawczasu (w trybie off-line) oraz druga grupa zadań prostszych realizowanych w czasie rzeczywistym w ramach zgłaszanych żądań treści. Po drugie, podział sieci CAN na domeny skutkuje koniecznością zastosowania routingu hierarchicznego. Oryginalną cechą naszego rozwiązania jest umiejętne połączenie wielu różnorodnych elementów: algorytmów optymalizacyjnych/heurystyk, wielościeżkowego routingu, a nawet wielokryterialnego rankingu, które skomponowane w spójny sposób zaowocowały mechanizmem zarówno realizowalnym jak i efektywnym.

W ramach routingu źródłowego sieci CAN pakiety treści są przesyłane przez tzw. węzły CAFE (ang. *Content Aware Forwarders*) na podstawie listy reguł dołączonej do pakietu w dostępowych węźle CAFE, który jest zlokalizowany w pobliżu serwera treści. Każdy węzeł CAFE kieruje pakiet do interfejsu wyjściowego, który określany jest na podstawie interfejsu wejściowego oraz reguły wyznaczonej przez wskaźnik następnej reguły (do tego celu CAFE używa odpowiednio skonfigurowanej tablicy). Węzeł przesuwa również wskaźnik następnej reguły na kolejną pozycję w liście reguł w pakiecie.

Routing źródłowy wymaga dosyć złożonej inżynierii ruchu, która pozwala na wyznaczenie listy reguł, a przez to całej trasy, przed rozpoczęciem przekazu treści. Mechanizm zarządzania ruchem ma w naszym przypadku charakter hierarchiczny ze względu na fakt podziału sieci CAN na domeny, które określane są również mianem systemów autonomicznych (ang. *Autonomous System*, AS), których topologia nie jest zwykle ujawniana innym domenom. Podstawowymi komponentami infrastruktury sieciowej domeny są węzły CAFE. Z zewnątrz domena jest widziana poprzez swoje węzły graniczne, którymi są węzły brzegowe CAFE, prowadzące do innych domen, lub węzły dostępowe CAFE, prowadzące do sieci dostępowych, a przez to również to serwerów treści jak i użytkowników sieci CAN.

Routing w sieci CAN jest realizowany w dwóch skalach czasowych. Mechanizm wyznaczający routing w długiej skali czasu jest uruchamiany zaraz po stworzeniu sieci CAN lub może być powtarzany w długich odstępach czasu, gdy topologia lub zapotrzebowanie ulegną znacznym zmianom. Proces wyznaczający

routing w krótkiej skali czasu, określany mianem fazy dynamicznej, jest wywoływany dla każdego żądania dostarczenia treści i wybiera najlepszą trasę spośród ścieżek wyznaczonych w ramach długiej skali czasowej. W krótkiej skali czasu uwzględniane jest aktualne obciążenie serwerów treści oraz stan sieci.

W długiej skali czasowej wykonywane są dwa poniższe rodzaje routingu:

1. *Routing wewnątrzdomenowy* - proces obliczeniowy, który jest wykonywany niezależnie w każdej domenie. Celem tego procesu jest wyznaczenie ścieżek w obrębie domeny, które przeniosą ruch pomiędzy każdą parą węzłów granicznych (od węzła wejściowego do węzła wyjściowego). Innymi słowy, routing wewnątrzdomenowy określa sposób przeniesienia ruchu sieciowego w ramach danej domeny widzianego z zewnątrz tej domeny.
2. *Routing międzydomenowy* - działa na poziomie domen i wyznacza ścieżki typu punkt-punkt między parami domen serwera treści i użytkownika. Pary takie definiują relacje w na poziomie międzydomenowym. Ogólnie rzecz biorąc, routing międzydomenowy wyznacza zbiór ścieżek dla danej relacji, uwzględniając przy tym zarówno wymagania na przesył treści, jak i zasoby przydzielone dla poszczególnych domen. Z tego względu routing międzydomenowy musi być wyznaczany po routingu wewnątrzdomenowym. Routing międzydomenowy jest realizowany w sposób rozproszony za pomocą specjalizowanego międzydomenowego protokołu. Protokół ten wykorzystuje algorytm wektora odległości oraz specjalizowaną funkcję kosztu ścieżki, która pozwala wybrać ścieżki preferowane ze zbioru ścieżek dopuszczalnych.

W krótkiej skali czasowej, dla każdego żądania treści, wykonywane są następujące operacje:

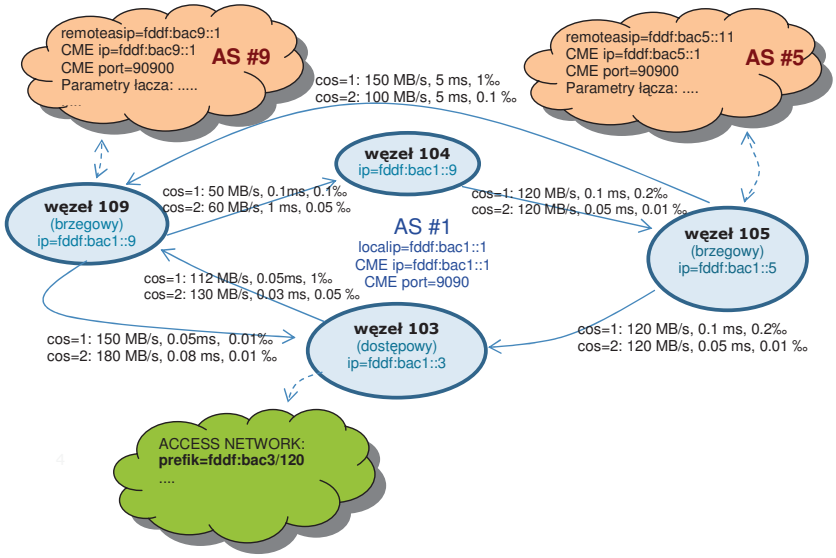
3. *Proces decyzyjny* - wybiera serwer źródłowy oraz ścieżkę przesyłu treści od tego serwera do użytkownika. Decyzja ta jest konieczna ze względu na podstawowe założenie sieci świadomych treści o tworzeniu kopii poszczególnych treści i umieszczaniu ich na różnych serwerach w celu równoważenia ich obciążenia. Jednakże proces decyzyjny jest luźno związany z samym routingiem i nie będzie omawiany w tym artykule. Więcej informacji można znaleźć na ten temat w [11].
4. *Konfiguracja ścieżki* - podstawowym celem tego modułu jest zestawienie ścieżki pomiędzy użytkownikiem, a serwerem treści wybranym przez proces decyzyjny spośród ścieżek wewnątrz- i międzydomenowych (listy domen). W ramach tego zadania abstrakcyjne opisy wybranych ścieżek w formie listy numerów domen przekształcane są w listy reguł, które wykorzystywane są w węzłach CAFE do przekazu treści. Z tworzeniem listy reguł wiąże się także konfiguracja węzłów dostępowych CAFE.

Struktura artykułu prezentuje się w następujący sposób. Rozdziały 2, 3 i 4 opisują odpowiednio routing wewnątrzdomenowy, routing międzydomenowy oraz konfigurację ścieżki. W rozdziale 4 cześć rozważań jest zilustrowana aktualnie przeprowadzanymi eksperymentami. W końcu rozdział 5 podsumowuje artykuł i przedstawia kierunki przyszłych prac.

## 2 Routing wewnątrzdomenowy

Wyliczenie routingu wewnątrzdomenowego, w dalszej części krócej, choć niezbyt ściśle nazywane także po prostu routingiem wewnątrzdomenowym ma miejsce w czasie wymiarowania domeny. Celem obliczeń jest wyznaczenie dla każdej *relacji* ścieżki wewnątrz domeny, którą będzie prowadzony ruch w tej relacji. Relacja to para różnych punktów z granicy domeny (tj. CAFE brzegowych lub dostępowych). Jeśli w relacji ma być realizowana więcej niż jedna klasa usług, dla każdej klasy w tej relacji wylicza się oddzielnie ścieżkę (choć niekoniecznie inną). Wymagamy, aby ścieżki spełniały szereg ograniczeń, jak spełnienie parametrów jakości usług QoS w relacjach.

Routing wewnątrzdomenowy zostanie zaprezentowany w swej esencjonalnej postaci, z użyciem abstrakcyjnego opisu grafowego domeny.




**Rysunek 1.** Przykładowa, uproszczona informacja o topologii domeny AS#1. Mamy $V = \{103, 104, 105, 109\}$, $E = \{(103, 109), (109, 103), (109, 104), (104, 105), (105, 109), (105, 103)\}$, $\mathrm{ASS}_{109}$ zawiera AS#9, $\mathrm{ASS}_{105}$ zawiera AS#5, $\mathrm{ANS}_{103}$ zawiera sieć dostępową o prefiksie fddf:bac3/120, pozostałe $\mathrm{ANS}_i$ i $\mathrm{ASS}_i$ są puste, $C = \{1, 2\}$, $\chi_{x,r,i}$ zostały wypisane przy łukach.

Właściwa implementacja troszczy się o dalsze szczegóły o charakterze technicznym, jak adresy IP różnych elementów sieci, usług, charakterystykami łączy międzydomenowych i in. Podstawe dane wejściowe dla routingu wewnątrzdomenowego są następujące:

1. Dane o topologii domeny (por. rys. 1). Najważniejszymi z nich są:
   (a) Zbiór węzłów (CAFE) $V \subseteq N$, zbiory $\mathrm{ANS}_i$ (dla $i \in V$) sieci dostępowych dołączonych do węzła $i$-tego, zbiory $\mathrm{ASS}_i$ (dla $i \in V$) innych systemów autonomicznych (domen) dołączonych do węzła $i$-tego. Poszczególne $\mathrm{ASS}_i$ oraz $\mathrm{ANS}_i$ mogą być puste. Elementy zbiorów $\mathrm{ANS}_i$ oraz $\mathrm{ASS}_i$ są złożonych typów, np. element zbioru $\mathrm{ASS}_i$ zawiera adresy IP różnyuch usług w sąsiedniej domenie, opis łącza między domeną naszą a sąsiednią i in.
   (b) Zbiór łączy (łuków) $E \subseteq V \times V$. Łącza w grafie są formalnie jednokierunkowe, dwukierunkowe łącza modelujemy jako dwa łącza w grafie.
   (c) Charakterystyki łaczy. Starano się tak sformułować zagadnienie routingu wewnątrzdomenowego aby uwzględnić charakterystyki łączy w różny sposób agregujące się przy łączeniu łączy w ścieżkę: albo agregujące się operacją minimum (jak przepustowość) albo sumowaniem (jak stopa strat pakietów, pod warunkiem, że jest dostatecznie mała i opóźnienie przekazu pakietów). Aktualnie charakterystykami łączy są $\chi_{c,e,i}$ dla $c \in C$, $e \in E$, $i \in K = \{0, \ldots, 2\}$. Liczby $\chi_{c,e,0}$ to przepustowości, $\chi_{c,e,1}$ to opóźnienia, $\chi_{c,e,2}$ to stopy strat. Zbiór $C \subseteq N$ to zbiór klas usług (Classes of Service – CoS). Jako że architektura systemu IIP przewiduje oddzielne kolejki dla poszczególnych klas usług, charakterystyki łączy są indeksowane przez klasę usług. Dla uproszczenia wywodów, podzieliliśmy także a priori pojemność łącza między klasy usług. W ten sposób otrzymujemy jednolity opis wszystkich charakterystyk, wszystkie trzy są tak samo indeksowane klasą usług. W rzeczywistości jednak to raczej od routingu wewnątrzdomenowego oczekujemy wyliczenia podziału zadanych całkowitych przepustowości łączy między klasy usług i dopiero po wykonaniu obliczeń przez moduł routingu wewnątzrzdomenowego chcemy rozdzielać przepustowości. Algorytm optymalizacji ma dostępne obydwie opcje, domyślną jest druga z nich, natomiast w tym artykule posługujemy się, dla prostoty, pierwszą.
   (d) Wydajności typu „PDU/s" dla węzłów – ograniczenia ograniczenie na całkowitą liczbę pakietów zsumowanych po wszystkich interfejsach wejściowych węzła, które procesor węzła jest w stanie obsłużyć w ciągu sekundy.
2. Wymagania na ruch realizowany w domenie.
   (a) Wymagane parametry $\breve{\chi}_{c,r,i}$ dla ścieżki w relacji $r$ obsługującej klasę usług $c$ – dla wszystkich możliwych relacji $r$ i $c \in C^r$ ($C^r$ to zbiór klas usług realizowanych w relacji $r$). Przyjęto, że $j = 0$ odnosi się do przepustowości, $j = 1$ – do opóźnienia, $j = 2$ – do stopy strat, jak wyżej.
   (b) Wymagania $\phi_{c,r}$ na natężenie ruchu w relacji $r$ dla klasy usług $c$ (dla $r$ i $c$ jak wyżej). Podczas gdy $\breve{\chi}_{c,r,0}$ opisuje wymaganie ruchowe dla pojedynczego połączenia w rozważanej relacji dla rozważanej klasy usług, odpowiednie $\phi_{c,r}$ opisuje wymaganie na zagregowany ruch generowany przez wiele jednoczesnych połaczeń w tej samej relacji i dla tej samej klasy usług. Oczywiście zazwyczaj $\breve{\chi}_{c,r,0}$ będzie mniejsze niż $\phi_{c,r}$. Nie oznacza to, że $\breve{\chi}_{c,r,0}$ jest nadmiarowe, gdyż ograniczenie na pojedyncze

połączenie musi być spełnione bezwarunkowo, jest ograniczeniem twardym, natomiast ograniczenie na ruch zagregowany jest ograniczeniem miękkim i może być nieco naruszone przez algorytm, jeśli nie może on uzyskać rozwiązania bez takich naruszeń.

Opisane wymagania są dla algorytmu danymi, specyfikowanymi raz przed wykonaniem obilczeń. Ich wyspecyfikowanie wymaga sporej dozy arbitralności. Jednym z prostszych sposobów ominięcia tej arbitralności jest ich ponowna specyfikacja i ponowne wyliczenie routingu wewnątrzdomenowego, gdy nasza wiedza o rzeczywistych wymaganiach stanie się pełniejsza, np. wskutek zgromadzenia większej ilości danych pomiarowych o ruchu. Jednak pełne i eleganckie rozwiązanie problemu określenie tych wymagań przyniosłaby dopiero hierarchiczna optymalizacja, ze współpracą modułów routingu wewnątrz- i zewnątrzdomenowego, co byłoby jednak niezwykle skomplikowane koncepcyjnie, zwłaszcza wobec faktu, iż routing międzydomenowy operuje zupełnie innym językiem od używanego przez routing wewnątrzdomenowy języka optymalizacji.

Wyjście algorytmu routingu wewnątrzdomenowego obejmuje:

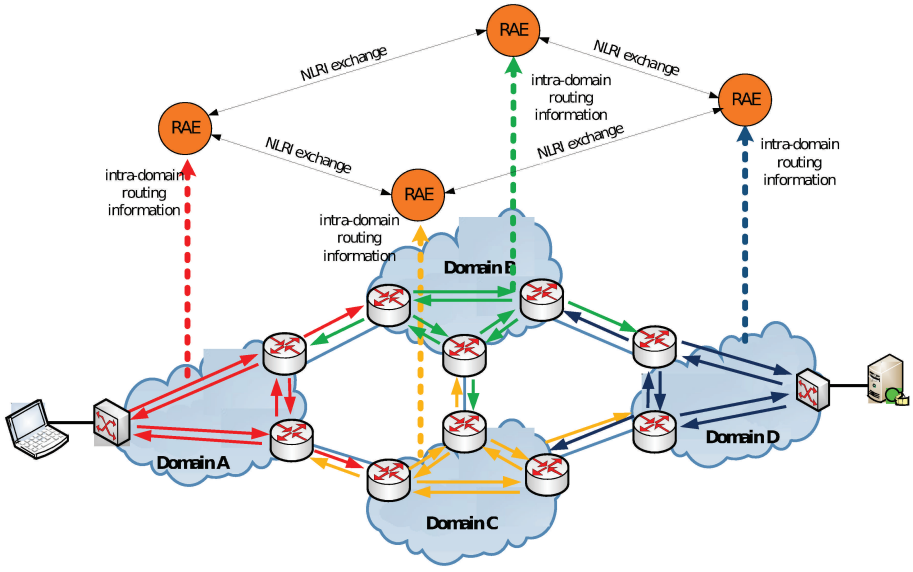
1. Po jednej ścieżce (ciągu elementów ze zbioru $V$) dla każdej możliwej pary relacja $r$ - klasa $c \in C^r$. Te dane są potrzebne procesowi konfiguracji ścieżki. Na podstawie szczegółowych danych topologicznych z systemu zarządzania CAN, obejmujących w szczególności interfejsy, ciąg węzłów jest tłumaczony na sekwencję tzw. kluczy forwardowania w pakiecie CAN – por. rozdział 4.
2. Charakterystyki (przepustowości, opóźnienia, straty) każdej ze ścieżek. Ta informacja, łącznie z danymi o węzłach z granicy domeny, jest przekazywana routingowi międzydomenowemu, który agregując tę informację z analogiczną informacją z innych domen wylicza ścieżki międzydomenowe.
   Zadanie obliczeniowe, jakim jest zadanie routingu wewnątrzdomenowego jest zapisane ściśle jako zadanie optymalizacji. Bazuje ono na zadaniu niepodzielnego przepływu wielotowarowego (zob. np. [12]), z towarami definiowanymi jako pary relacja - klasa usług. Zadanie cechuje się bardzo zawiłym sformułowaniem, uwzględniającym liczne szczegóły, jak agregację charakterystyk łaczy wzdłuż ścieżek i zwielokrotnianie szeregu zmiennych i równań wynikłe z indeksowania ich klasą usług, cechuje je także znaczna złożoność obliczeniowa. Praktyczne rozwiązanie tego zadania wydawało się możliwe jedynie na drodze heurystycznej. Algorytm jest zatem heurystyką składającą się z dwóch głównych elementów:
   (a) Poszukiwania kilku ścieżek dla pary relacja - klasa usług, ścieżki powinny być dopuszczalne w sensie przepustowości (dla pojedynczego połączenia, ignorujemy chwilowo istnienie innych ścieżek i innych jednoczesnych połączeń), opóźnień i stóp strat. Przy tym ograniczenia na pojemność są uwzględniane przez proste odrzucenie łuków grafu o zbyt małych przepustowościach, natomiast ograniczenia na opóźnienie i stopę strat są uwzględniane przez specjalny dwukryterialny algorytm najkrótszej ścieżki, który z kolei jest sprowadzany do zwykłego, jednokryterialnego algorytmu najkrótszej ścieżki przez ważenie i parametryzację i ranking wielokryterialny metodą poziomów odniesienia ([14], rozdz. 4). Jest to

jedno z prostszych możliwych wielokryterialnych podejść do problemu najkrótszej ścieżki [13]. Wydaje się jednak praktycznie uzasadnione: jest tanie i daje przybliżone rozwiązania, uzasadnione, gdy założymy pewien poziom przewymiarowania dostępnych zasobów. Nadto podejście nasze umożliwia łatwe wprowadzenie kar w zewnętrznej pętli algorytmu.

(b) Pętli zewnętrznej, która bierze pod uwagę zajmowania przepustowości łączy przez wiele ścieżek jednocześnie (także ograniczenia węzłów typu „PDU/s."). Pętla ta używa kar za przekroczenie ograniczeń. Kary są dodawane do kosztów łączy w trakcie wyznaczania najkrótszych ścieżek.

## 3 Routing międzydomenowy

Routing międzydomenowy polega na wyliczeniu ścieżek (tras) międzydomenowych. Routing międzydomenowy jest przeprowadzany w dużej skali czasowej. Może on też uwzględnić zmiany w osiągalności sieci lub zmiany w routingu wewnątrzdomenowym. Routing międzydomenowy jest przeprowadzany w modułach Routing Awareness Entity (RAE) zlokalizowanych w każdej z domen CAN, co zaprezentowano na rys. 2.




**Rysunek 2.** Routing wewnątrzdomenowy w architekturze sieci CAN.

RAE implementuje dwie nowe funkcjonalności: (1) routing wielokryterialny (QoS), który pozwala zbudować trasy międzydomenowe przy uwzględnieniu wymagań na transfer treści (2) routing wielościeżkowy, gdyż RAE buduje cały

zestaw ścieżek wiodących do danego prefiksu. Każde RAE otrzymuje informację o swej domenie od routingu wewnątrzdomenowego. Ta informacja obejmuje wartości przepustowości, opóźnienia i stopy strat pakietów – oferowane dla każdej z relacji wewnątrzdomenowych, zdefiniowanej parami punktów wejściowych i wyjściowych. Nadto, RAE otrzymuje adresy RAE z sąsiednich domen. Następnie, podobnie jak w protokole BGP-4, RAE rozpoczyna wymianę wiadomości WITHDRAW i UPDATE z innymi RAE zlokalizowanymi w sąsiednich domenach aby na tej podstawie zbudować ścieżki międzydomenowe. W wypadku wykrycia zmiany sytuacji, takiej jak awaria w sąsiedniej domenie lub zmiana informacji od routingu wewnątrzdomenowego informacja w routingu międzydomenowym podlega uaktualnieniu i rozpropagowaniu w całej sieci.

Każde RAE utzymuje dwie tablice routingu: Tablicę Tras Znanych – Known Routes Tables (KRT) oraz Tablicę Tras Preferowanych – Preferred Routes Table (PRT). Pierwsza przechowuje informacje o trasach oferowanych przez sąsiednie RAE, druga natomiast – o trasach preferowanych przez domenę. RAE używa algorytmu rankingu tras do wyznaczania wszystkich znanych tras i wyznaczenia zestawu tras preferowanych. W podejściu wielościeżkowym algorytm rankingu tras wyznacza kilka preferowanych tras do ustalonego prefiksu.

Skoncentrujmy uwagę na operacjach RAE, tj. opearcjach przeprowadzanych przez RAE po otrzymaniu wiadomości UPDATE, WITHDRAW lub KEEP-ALIVE.

1. Kiedy RAE otrzymuje wiadomość UPDATE od sąsiedniej domeny:
   (a) Przeprowadza test poprawności wiadomości UPDATE eliminując trasy zawierające jej własny numer AS. Proces ten pozwala uniknąć zapętleń.
   (b) Uaktualnia własności tras odebranych w wiadomości UPDATE, tj. długość ścieżki, lista domen, zagregowane parametry QoS.
   (c) Wywołuje algorytm rankingu tras, który wybiera nowe trasy preferowane i przygotowuje nowe PRT.
   (d) Jeśli nowe PRT jest identyczne z poprzednim, proces się kończy. W przeciwnym wypadku RAE uaktualnia PRT i wysyła do sąsiadów wiadomość UPDATE z aktualnymi trasami preferowanymi. Proces się kończy.
2. Kiedy RAE otrzymuje wiadomość WITHDRAW z sąsiedniej domeny:
   (a) Usuwa trasy z KRT.
   (b) Wywołuje algorytm rankingu tras w celu przygotowania nowego PRT.
   (c) Jeśli nowe PRT jest identyczne z dotychczasowym, proces się kończy. W przeciwnym razie RAE uaktualnia PRT i wysyła do swych sąsiadów wiadomość WITHDRAW zawierającą zbiór tras do usunięcia. Proces się kończy.
3. Kiedy RAE otrzymuje wiadomość KEEP-ALIVE od sąsiedniej domeny:
   (a) Odświeża licznik timeoutu (*keep_alive*)

Należy zwrócić uwagę, iż kiedy licznik *keep_alive* przeterminowuje się, RAE zaczyna traktować sąsiada jako niedostępnego i w związku z tym:

(a) Usuwa z KRT wszystkie trasy otrzymane z tej pary.
(b) Wywołuje algorytm rankingu tras w celu przygotowania nowego PRT.

(c) Jeśli nowe PRT jest identyczne z poprzednim, proces się kończy. W przeciwnym wypadku RAE uaktualnia PRT i wysyła do sąsiadów wiadomość ze zbiorem tras do usunięcia. Proces się kończy.

## 4 Konfiguracja ścieżki

W wyniku dokonanego procesu wyszukiwania treści (*content resolution*) system CAN zna: adres IP węzłów dostępowych, zwanych Content Delivery Proxy (CDP), zarówno po stronie serwera jak i konsumenta treści, jak również ścieżkę (w formie listy numerów AS), która będzie używana do dostarczania treści. W następnym kroku CAFE w sieci CAN musi zostać przygotowane do dostarczenia treści od serwera treści do klienta. W tym celu uruchamiany jest proces konfiguracji ścieżki.

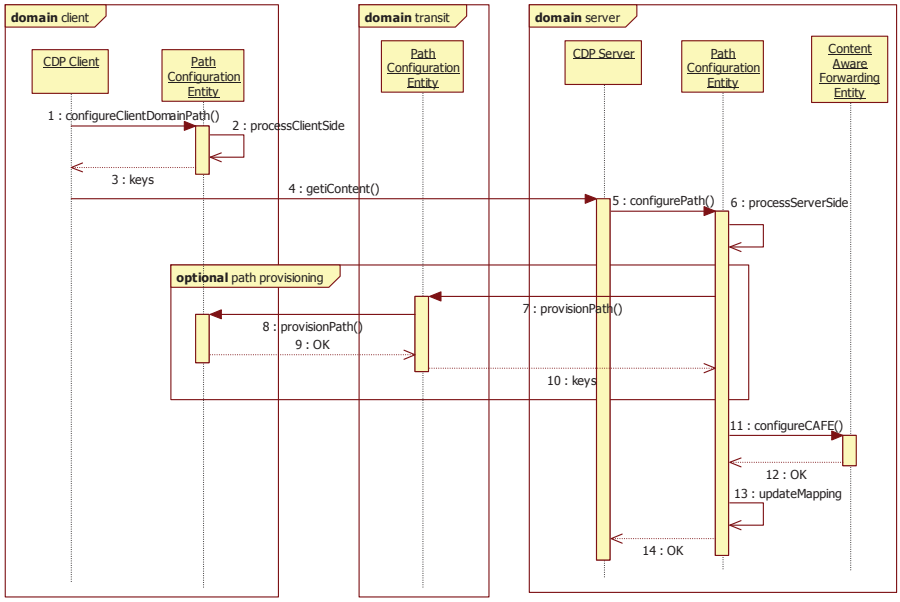
Proces konfiguracji ścieżki obejmuje dwa główne działania:

1. Tłumaczenie wybranej ścieżki, mającej formę listy numerów AS na listę wskaźników reguły forwardowania (kluczy forwardowania). Kluczy tych używają poszczególne CAFE do zastosowania reguły forwardowania (przełączania) dla przesyłanych pakietów.
2. Konfigurację CAFE zlokalizowanego najbliżej serwera CDP tak aby przechwytywała pakiety IP i enkapsulowała z dodaniem nagłówka CAN zawierającego listę kluczy forwardowania.

Na diagramie z rys. 3 przedstawiono przepływ dla przypadku gdy serwer treści i klient znajdują się w różnych domenach.

Wymiana wiadomości ma początek po stronie klienta podczas gdy konfigurowana ścieżka ma prowadzić od serwera treści do klienta. Dlatego w pierwszym etapie Path Configuration Entity (PCE) znajduje klucze używane do przełączania pakietu od wejściowego CAFE domeny klienta do CDP klienta. Następnie klient CDP nawiązuje połączenie z CDP po stronie serwera, które kontaktuje się ze swym PCE. Wiadomość wysyłana pomiędzy CDP zawiera ścieżkę (ciąg) numerów AS, adres IP CDP klienta i listę kluczy.

Celem procesu zachodzącego po stronie serwera jest znalezienie kluczy dla przełączania pakietu od adresu CDP serwera do wejściowego CAFE domeny klienta, być może poprzez domeny tranzytowe. W procesie tym wyróżniamy dwa podprocesy. Pierwszy, nazwany procesem wyznaczania ścieżki (Path Provisioning Process), jest wywoływany tylko gdy brak jest odpowiedniej ścieżki wśród przechowywanych w bazie danych PCE. Podproces ten odpowiada za znalezienie kluczy używanych do pakietu od wyjściowego CAFE domeny serwera do wejściowego CAFE domeny klienta. Na diagramie założono brak odpowiedniej ścieżki w bazie danych, dlatego PCE od strony serwera wywołuje podproces wyznaczania ścieżki. Sytacja taka zachodzi gdy żądana ścieżka nie została do tej pory skonfigiurowana. Jest to w zasadzie rzadki przypadek, zachodzący jedynie dla pierwszego żadania wzdłuż danej ścieżki. Należy zwrócić uwagę na możliwość statycznego skonfigurowania ścieżek zgodnie z uprzednio zawartą umową



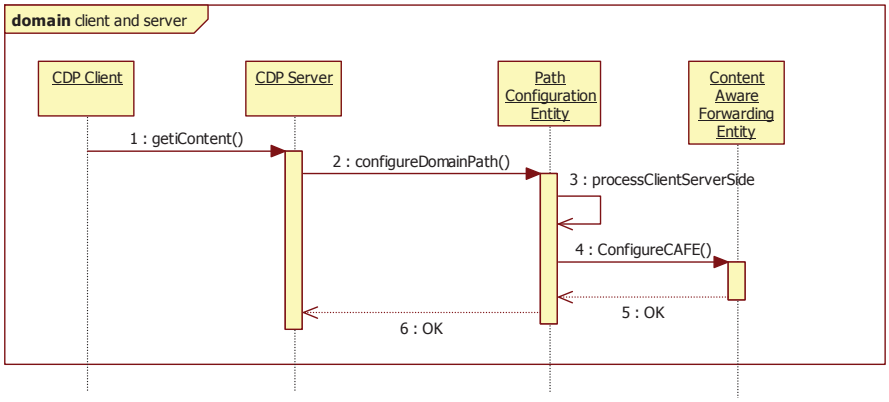

**Rysunek 3.** Proces konfiguracji ścieżki.

pomiędzy dostawcami usług sieciowych. Drugi podproces odpowiada za znalezienie kluczy używanych do przełączania pakietu od CAFE serwera do wyjściowego CAFE serwera domeny.

W następstwie powyższego PCE po stronie serwera otrzymuje listę wskaźników (kluczy) reguły forwardowania które tworzą ścieżkę od CDP serwera do CDP klienta. W końcowym kroku PCE wysyła żądanie konfiguracji do CAFE połączonego z CDP po stronie klienta. Żądanie zawiera adresy IP CDP klienta i serwera, oznaczenie protokołu transportowego, numery portów, żądaną przepływność, listę kluczy oraz czas odświeżania oznaczający ważność stanu tymczasowego w CAFE (każdy pakiet wysłany przez serwer do klienta odświeża go). Każde CAFE ma własną tabelę forwardowanaia dla przełączania pakietu do odpowiedniego interfejsu z użyciem reguły forwardowania specyficznej dla wykorzystywanej technologii. Forwardowanie opiera się na wartości kluczy zawartych w nagłówku pakietu utworzonym w CAFE po stronie serwera.

Wszystkie wyżej opisane procesy zlokalizowane w PCE wykorzystują informację dostarczaną przez proces zarządzania CAN i przechowywaną w trzech tablicach. Pierwsza z nich służy do znajdowania dostępowego CAFE odpowiedzialnego za obsługę ruchu do/z danego adresu IP. Jest ona potrzebna jedynie w domenie serwera lub klienta. Druga zawiera informację o CAFE transferujących (lub odbierających) pakiety pomiędzy sąsiednimi domenami. Trzeciej używa się do znajdowania klucza forwardowania, który pozwala forwardować pakiety między sąsiadujacymi CAFE. Poza wymienionymi trzema, każde PCE

ma czwartą tabelę ze zbiorem aktywnych ścieżek od sieci dostępowej do sieci dostępowej.




**Rysunek 4.** uproszczony proces konfiguracji ścieżki.

Powyżej opisane może zostać uproszczone w przypadku gdy serwer treści i klient znajdują się w tej samej domenie (zob. rys. 4).

## 5 Eksperymenty

Przeprowadzono eksperymenty dla zilustrowania omawianych zagadnień. Eksperymenty dotyczą routingu wewnątrzdomenowego. Poniżej podano kilka przykładów dla topologii sieci z rys. 1. Dla dwóch różnych ustawień wymagań wykonano routing wewnątrzdomenowy. We wszystkich możliwych relacjach wymagamy realizacji dwóch klas usług, które oznaczymy numerami 1 i 2, z wyjątkiem relacji AS#5↦AS#9, gdzie wymagamy jedynie klasy usług o numerze 1. Innymi słowy, przyjęto $C_r = \{1, 2\}$ dla wszystkich relacji $r$ z wyjątkiem relacji AS#5↦AS#9, dla której zachodzi $C_{AS\#5 \mapsto AS\#9} = \{1\}$. Użyliśmy zrozumiałej, nie wymagającej wyjaśnień notacji dla relacji. Wymagania przedstawiono w tabeli 1. Ograniczeń wydajności węzłów określanej w PDU/s nie uwzględniono w eksperymentach.

Wyniki obliczeń przedstawiono w tabeli 2.

Dla obydwu zestawów wymagań uzyskaliśmy rozwiązania dopuszczalne. Wymagania w eksperymencie B różnią się od wymagań w eksperymencie A mniejszym dopuszczalnym opóźnieniem w relacjach dla CoS #1: 2 ms wobec 7 ms, nadto, dla kompensacji – mniejszym natężeniem ruchu: 20MB/s wobec 100 MB/s. W rezultacie ścieżka dla AS#5↦AS#9 / CoS #1, która w eksperymencie A prowadziła bezpośrednio łączem (105,109) nie mogła pozostać niezmieniona w eksperymencie B, gdyż nie spełniłaby wymagania na opóźnienie i tym samym stałaby sie niedopuszczalna. Algorytm wybrał natomiast dłuższą ścieżkę

(105,103,109) na której łącze ma mniejsze opóźnienia. Przy zredukowanym w wymaganiach eksperymentu B wymaganiach na natężenie ruchu, otrzymane rozwiązanie było dopuszczalne zarówno w sensie opóźnienia jak i przepustowości ścieżek. Czasy obliczeń były satysfakcjonujące, to samo można powiedzieć o liczbie wywołań podprogramu obliczania najkrótszej ścieżki i liczby iteracji pętli zewnętrznej (pętli funkcji kary).

| **Experyment** | **Wymagania (wszystkie relacje $r$)** | | | | |
|---|---|---|---|---|---|
| | **CoS** ($c$) | **Ruch [MB/s]** ($\phi_{c,r}$) | **Przepustowość [MB/s]** ($\bar{\chi}_{c,r,0}$) | **Opóźnienie [ms]** ($\bar{\chi}_{c,r,1}$) | **Straty [‰]** ($\bar{\chi}_{c,r,2}$) |
| **A** | 1 | 100 | 20 | 7 | 3 |
| | 2*) | 60 | 30 | 2 | 1 |
| **B** | 1 | 20 | 20 | 2 | 3 |
| | 2*) | 60 | 30 | 2 | 1 |

**Tablica 1.** Wymagania dla eksperymentów. *) – z wyj. relacji AS#5 $\mapsto$ AS#9.

## 6 Podsumowanie i wnioski

Podejście do routingu przyjęte w naszej sieci CAN zapewnia znaczącą kontrolę nad ruchem. Problem routingu dekomponuje się bowiem na kilka podproblemów, w szczególności podproblem routingu wewnątrzdomenowego z potencjałem bardzo precyzyjnego planowania ruchu wewnątrz domen, podproblem routingu międzydomenowego charakteryzującego się elastycznością wynikającą z definiowania kilku ścieżek międzydomenowych dla pary relacja-CoS, z których proces decyzyjny wybierze jedną, kierując się aktualnym obciążeniem łączy i serwerów. Wstępne eksperymenty wskazują na poprawność prezentowanego podejścia. Hierarchiczny charakter routingu stanowi sposób zapewnienia skalowalności rozwiązania. Ceną, jaką płacimy jest zwiększona złożoność rozwiązania, zarówno w sensie złożonosci poszczególnych podproblemów i organizacji całej implementacji, w tym budowy fragmentów oprogramowania odpowiedzialnych za konfigurację ścieżki. Ponadto hierarchiczny routing, z bardzo różniącymi się od siebie algorytmami w poszczególnych warstwach jest koncepcyjnie trudny, złożony i wymaga stosowania uproszczeń w kilku miejscach, na przykład mniej lub bardziej arbitralnych ustawień wymagań dla routingu wewnątrzdomenowego. Musimy sobie zdawać sprawę z tego, iż w sieci budowanej na na zasobach wirtualnych dochodzi jeszcze jedna warstwa w hierarchii – związana z wymiarowaniem sieci dokonywanym podczas fazy wirtualizacji zasobów. Indukuje to problem routingu i zagadnienie koordynacji warstw o takiej złożonosci, że można je dziś śmiało okreslić jako ważne, otwarte problemy badawcze.

| Experyment | Relacja | Klasa usług (CoS) | Ścieżka | Pojemność ścieżki [MB/s] | Opóźnienie ścieżki [ms] | Straty ścieżki %] | Czas obliczeń [s] | Liczba wywołań algorytmu najkrótszej ścieżki | Iteracje pętli zewnętrznej | Niedopuszczalność ruchu [Mb/s] |
|---|---|---|---|---|---|---|---|---|---|---|
| **A** | AS#9 → AS#5 | 1 | (109,104,105) | 100 | 0.2 | 0.3 | 0.2 | 270 | 9 | 0 |
| | | 2 | (109,104,105) | 60 | 1.05 | 0.06 | | | | |
| | AS#9→ fddf:bac3::0/120 | 1 | (109,103) | 100 | 0.05 | 0.01 | | | | |
| | | 2 | (109,103) | 60 | 0.08 | 0.01 | | | | |
| | AS#5→ AS#9 | 1 | (105,109) | 100 | 5 | 1 | | | | |
| | AS#5→ fddf:bac3::0/120 | 1 | (105,103) | 100 | 0.1 | 0.2 | | | | |
| | | 2 | (105,103) | 60 | 0.05 | 0.01 | | | | |
| **B** | AS#9→ AS#5 | 1 | (109,104,105) | 20 | 0.2 | 0.3 | 0.19 | 270 | 9 | 0 |
| | | 2 | (109,104,105) | 60 | 1.05 | 0.06 | | | | |
| | AS#9→ fddf:bac3::0/120 | 1 | (109,103) | 20 | 0.05 | 0.01 | | | | |
| | | 2 | (109,103) | 60 | 0.08 | 0.01 | | | | |
| | AS#5→ AS#9 | 1 | (105,103,109) | 20 | 0.15 | 1.2 | | | | |
| | AS#5→ fddf:bac3::0/120 | 1 | (105,103) | 20 | 0.1 | 0.2 | | | | |
| | | 2 | (105,103) | 60 | 0.05 | 0.01 | | | | |

**Tablica 2.** Wyniki eksperymentów. Miara „niedopuszczalność ruchu" oznacza zsumowane niedopuszczalności dla wszystkich łączy i klas usług. Z kolei niedpuszczalność dla danego łącza i klasy usług oznacza przekroczenie pojemności łącza dla danej klasy usług przez zsumowane pojemności ścieżek biegnących przez to łącze (lub zero, gdy przekroczeń brak).

## Literatura


1. EC FIArch Group, Fundamental Limitations of Current Internet and the path to Future Internet, report ver. 0.9, 2010, `http://www.future-internet.eu/uploads/%20media/FIArch_Current_Internet_Limitations_V0_9.pdf`.
2. Koponen, T. et al. A Data-oriented (and Beyond) Network Architecture. Proc. ACM SIGCOMM '07, Kyoto, Japonia, Aug. 2007.
3. Jacobson, V., et al. Networking Named Content. Proc. ACM CoNEXT 09, 2009, pp.1-12.
4. Aranda, P., A. et al. Final Architectural Framework, June 2010. `http://www.4ward-project.eu/`.
5. Scalable and Adaptive Internet Solutions (SAIL) project. `http://www.sail-project.eu/`.
6. Chai, W. K. et al. Final specification of mechanisms, protocols and algorithms for the mediation system. COMET Deliverable D3.2, January 2012.
7. Chai, W. K. et al. CURLING: Content-ubiquitous resolution and delivery infrastructure for next generation services, IEEE Communications Magazine, vol. 49, no. 3, 2011, pp. 112-120.
8. W. Burakowski, H. Tarasiuk, A. Bęben. Architektura Systemu IIP. Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011).
9. A. Bęben i in. Architektura sieci świadomych treści w systemie IIP. Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011).

10. Bęben, A. et al. Content Aware Network based on Virtual Infrastructure. Proc. 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2012), Kyoto, Japan, August 2012.
11. Bęben, A., Batalla, J. Chai, W. K. and Śliwiński, J. Multi-criteria decision algorithms for efficient content delivery in content networks. Annals of Telecommunications, Special Issue on Networked digital media, Springer, 2012 (to appear).
12. Ahuja, R., Magnanti, T. L., Orlin, J. Network Flows: Theory, Algorithms and Applications., Prentice-Hall. 1993, Upper Saddle River, NJ, USA.
13. Garroppo, R., Giordano, S., Tavanti, L. A survey of multi-constraint optimal path computation: Exact and approximate algorithms. Computer Networks 54 (2010), pp. 3081-3107.
14. Wierzbicki, A. P., Wessels, J. (eds.) Model-based decision support methodology with environmental applications. Kluwer, Dodrecht, 2000.

# Aplikacje Sieci Świadomej Treści

Michał Kosiedowski[1], Ewa Kuśmierek[1], Aleksander Stroiński[1], Michał Cholewa[2], Przemysław Głomb[2], Sebastian Opozda[2], Michał Romaszewski[2], Arkadiusz Sochan[2], Andrzej Bęben[3], Piotr Krawiec[3]

[1] Poznańskie Centrum Superkomputerowo Sieciowe
[2] Instytut Informatyki Teoretycznej i Stosowanej Polskiej Akademii Nauk
[3] Instytut Telekomunikacji, Politechnika Warszawska

**Streszczenie** Artykuł przedstawia aplikacje korzystające z sieci świadomej treści PI CAN opracowanej w ramach projektu Inżynieria Internetu Przyszłości (IIP). Są to trzy aplikacje oferujące dostęp do treści multimedialnych: HomeNetMedia dla celów rozrywki, Rozproszone Wirtualne Muzeum dla edukacji oraz Biblioteka Cyfrowa Pacjenta dla przekazywania informacji medycznej. Scharakteryzowana została funkcjonalność sieci PI CAN z punktu widzenia aplikacji wraz z zasadami współpracy z użytkownikami, tj. dostawcami i konsumentami treści. Poszczególne aplikacje omówione zostały ze wskazaniem sposobu wykorzystania sieci świadomej treści oraz implementacji usług zewnętrznych powiązanych z siecią PI CAN specyficznych dla aplikacji. Przedstawione zostały również wyniki integracji poszczególnych aplikacji z siecią PI CAN oraz wyniki przeprowadzonych testów.

## 1 Wprowadzenie

W ostatnim czasie zaobserwować można rosnącą popularność usług oferujących treści multimedialne, takie jak filmy wideo, utwory muzyczne, programy radiowe i telewizyjne, z wykorzystaniem szerokopasmowych sieci IP. Nowa platforma dystrybucji multimediów coraz częściej wygrywa z tradycyjnymi platformami dystrybucji, ponieważ oferuje nowe możliwości i wartości dodane dla usług z niej korzystających. Z jednej strony nowe możliwości stanowią o atrakcyjności usług multimedialnych, z drugiej jednak nowa platforma musi również sprostać oczekiwaniom użytkowników przeniesionym z tradycyjnych platform, a dotyczących krótkiego czasu dostępu i realizacji żądań użytkownika oraz jakości przekazu. Dotyczy to nie tylko treści oferowanych w celach rozrywkowych, ale również dla celów edukacyjnych, popularyzacji wiedzy czy przekazywania informacji medycznej, której swobodna wymiana jest niezwykle istotna w scenariuszach z obszaru eZdrowia. Platformy klasy EHR (ang. *Electronic Health Record*), umożliwiające współdzielenie dokumentacji medycznej w postaci cyfrowej pomiędzy rozproszonymi lokalizacjami, są obecnie jednym z głównych obszarów innowacji wprowadzanej w Służbie Zdrowia [1].

Przed dostawcą usług multimedialnych stoją więc dwa wyzwania. Pierwsze związane jest z oczekiwaniami użytkowników dotyczącymi m.in. prostego i intuicyjnego mechanizmu wyszukiwania treści i ich pobierania. Proces ten powinien być transparenty dla użytkownika bez konieczności znajomości z góry identyfikatora treści, jej lokalizacji, czy adresu serwera, który ją udostępnia. Użytkownik powinien być również zwolniony z konieczności zdawania sobie sprawy z istnienia wielu różnych niezależnych katalogów treści i ich przeszukiwania. Powinien za to mieć możliwość odtworzenia wybranej treści na urządzeniu, które ma w danym momencie pod ręką.

Drugie wyzwanie związane jest z zapewnieniem wysokiej jakości usługi odbieranej przez użytkownika QoE (ang. *Quality of Experience*), co wiąże się m.in. z zapewnieniem wysokiej jakości QoS (ang. *Quality of Service*) przez sieć transmitującą treści. Dostawcy treści z kolei oczekują mechanizmów umożliwiających udostępnienie wysokiej jakości treści użytkownikom na określonych przez dostawcę zasadach wynikających z charakteru treści i celu jej wykorzystania. Niektóre treści mogą wymagać szczególnej ochrony nie tylko ze względu na ochronę praw autorskich, ale również w przypadku, gdy są to np. dane medyczne pacjenta i dostęp do nich powinien być ograniczony do wskazanej grupy użytkowników. Trudne, jeśli nie wręcz niemożliwe, jest spełnienie tych wymagań wykorzystując Internet z usługą typu „Best Effort" jako medium transmisji treści multimedialnych. Jego ograniczenia przyczyniły się do rozpoczęcia prac nad nową architekturą sieci, tzw. sieci świadomej treści - CAN (ang. *Content Aware Network*), która jest specjalizowana dla przekazu treści. Sieć taka powstała w ramach projektu Inżynieria Internetu Przyszłości [2] jako jeden z równoległych Internetów i określana jest mianem PI CAN (ang. *Parallel Internet CAN*) [3].

W niniejszym artykule przedstawiona jest funkcjonalność sieci PI CAN z punktu widzenia użytkowników tzn. dostawców treści oraz konsumentów treści. Zaprezentowane są trzy aplikacje wykorzystujące sieć PI CAN do realizacji swoich usług: HomeNetMedia, Rozproszone Wirtualne Muzeum oraz Biblioteka Cyfrowa Pacjenta. Aplikacje te udostępniają treści multimedialne odpowiednio dla rozrywki, edukacji oraz eZdrowia. Stanowią one przykład z jednej strony wykorzystania możliwości jakie oferuje sieć ukierunkowana na przekaz treści multimedialnych, z drugiej zaś przykład aplikacji i usług, które definiują nowe wymagania w stosunku do sieci wykraczające poza usługi transmisji danych. Część z tych wymagań, jest wspólna dla wszystkich aplikacji, część jednak jest specyficzna i wynika z charakteru oferowanych przez nie usług.

W rozdziale 2 przedstawiona została sieć PI CAN i jej funkcjonalność z punktu widzenia dwóch grup użytkowników. Opis dotyczy zarówno funkcji realizowanych niezależnie od wykorzystujących je aplikacji, jak i funkcji, których implementacja jest specyficzna dla aplikacji. W rozdziale 3 zaprezentowane są poszczególne aplikacje ze wskazaniem sposobu wykorzystania usług sieci PI CAN oraz implementacji funkcji specyficznych dla danej aplikacji. Rozdział 4 zawiera opis integracji aplikacji z siecią PI CAN wraz z uzyskanymi wynikami przeprowadzonych testów. Podsumowanie przedstawione jest w rozdziale 5.

## 2 Sieć świadoma treści

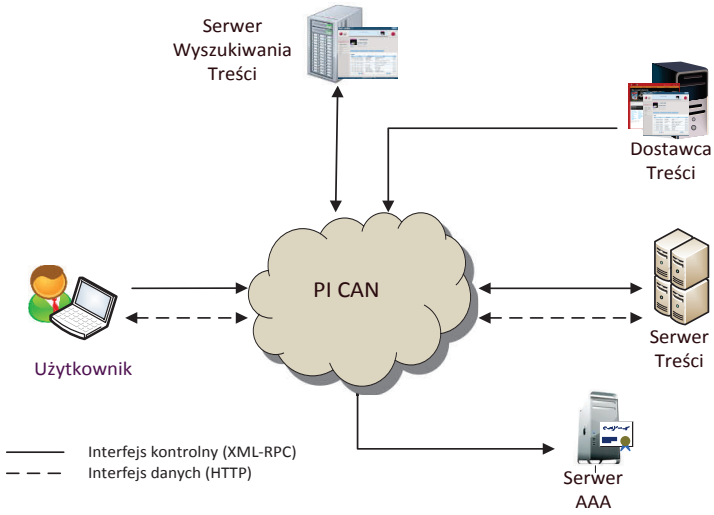
Sieć PI CAN jest siecią ukierunkowaną na przekaz treści multimedialnych. Realizuje ujednolicony dostęp do treści, schemat nadawania identyfikatorów treści oraz metodę przechowywania i wyszukiwania informacji o treści, mechanizmy sterowania siecią i pozyskiwania informacji o obciążeniu serwerów i ruchu w sieci, oraz proces decyzyjny umożliwiający wybór optymalnego serwera i ścieżki w sieci, które zostaną użyte do obsługi żądania wysłanego przez użytkownika.

Grupa użytkowników sieci PI CAN obejmuje dostawców treści oraz odbiorców końcowych czyli konsumentów treści. Z punktu widzenia użytkowników PI CAN realizuje dwie podstawowe operacje, tj. publikowanie treści oraz pobieranie treści. Publikowanie jest realizowane przez dostawców treści i ma na celu udostępnienie treści użytkownikom sieci PI CAN. Pobieranie treści inicjuje użytkownik przez wysłanie do sieci PI CAN żądania pobrania treści.

Publikacja treści przebiega w trzech etapach. Pierwszy etap polega na umieszczeniu treści na tzw. serwerach treści. Drugi to zarejestrowanie treści polegające na przekazaniu informacji o lokalizacji replik treści na serwerach, profilu ruchowego określającego szczytową szybkość transmisji (ang. *peak rate*) oraz wymagań dotyczących jakości przekazu treści przez wskazanie klasy usług (ang. *Class of Service*) i zarejestrowaniu ich wraz z unikalnym identyfikatorem przydzielanym treści. Profil ruchu oraz klasa usług określają wymagania w odniesieniu do transmisji treści i stanowią podstawę wyboru ścieżki. Dodatkowe informacje przekazywane na tym etapie przez dostawcę obejmują wskazanie czy dana treść może być cache'owana i czy dostęp do niej wymaga autoryzacji użytkownika. Trzeci etap to zarejestrowanie opisu, czyli meta danych treści, wykorzystywanego później w procesie wyszukiwania treści.

Pobranie treści wymaga znajomości identyfikatora treści oraz adresu węzła sieci PI CAN, do którego należy wysłać żądanie. W związku z tym pobranie treści przez użytkownika jest zwykle poprzedzone wyszukaniem treści na podstawie podanych przez użytkownika słów kluczowych. Wynikiem pomyślnie zakończonej operacji wyszukania treści są metadane treści spełniających podane kryteria wyszukiwania oraz adres, na który należy wysłać żądanie wydania treści. Odwołanie się do uzyskanego adresu uruchamia cały proces wyboru serwera oraz ścieżki transmisji i w efekcie dostarczenie treści do użytkownika końcowego. Zarówno operacja publikacji treści, jak i wyszukania oraz pobrania treści może wymagać uwierzytelnienia i autoryzacji użytkownika, odpowiednio dostawcy i konsumenta treści.

Realizacja operacji takich jak wyszukiwanie treści czy autoryzacja użytkownika zależy w dużym stopniu od aplikacji wykorzystującej PI CAN. Treści mogą bowiem być opisywane w bardzo różny sposób, zależny od ich rodzaju treści i celu wykorzystania. Różne mogą być zatem kryteria wyszukiwania. Podobnie, proces weryfikacji uprawnień użytkownika do pobrania treści jest zależny od aplikacji. W związku z tym wyszukiwanie identyfikatora treści, uwierzytelnianie, autoryzacja i naliczanie oraz przechowywanie źródłowych kopii treści, są realizowane przez usługi zewnętrzne. Są one oferowane odpowiednio przez Serwery




**Rysunek 1.** Komponenty aplikacji HomeNetMedia

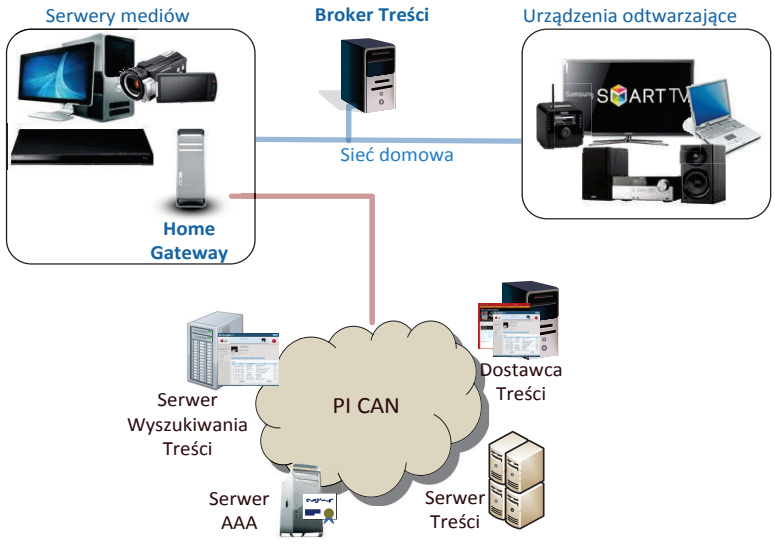
Wyszukiwania Treści (ang. *Content Search Server*, CSS), Serwery AAA (ang. *Authentication, Authorization, Accounting*, AAAS) oraz Serwery Treści (ang. *Content Server*, CS). Serwery te muszą implementować odpowiednie interfejsy do komunikacji z komponentami sieci CAN.

Sieć PI CAN udostępnia użytkownikom dwa interfejsy: kontrolny i przekazu danych. Rys. 1 przedstawia wykorzystanie obu interfejsów przez użytkowników oraz usługi zewnętrzne. Interfejs kontrolny pozwala dostawcy treści na autoryzację, zarządzanie danymi treści oraz zarzadzanie metadanymi opisowymi treści. Zarządzanie obu rodzajami danych obejmuje operacje rejestracji, modyfikacji oraz usunięcia danych. Konsument treści korzysta z funkcji uwierzytelniania oraz wyszukania treści na podstawie podanych słów kluczowych. Dla obu grup użytkowników wynikiem pomyślnego uwierzytelnienia jest uzyskanie żetonu (ang. *token*) wykorzystywanego następnie przy wywołaniu pozostałych funkcji.

W obecnej implementacji sieci PI CAN, przekazanie treści do sieci przez dostawcę lub pobranie jej przez konsumenta realizowane jest za pomocą protokołu HTTP, odpowiednio metod PUT i GET.

## 3 Usługi PI CAN w aplikacjach

W rozdziale tym przedstawiona została charakterystyka każdej z trzech aplikacji z opisem usług jakie oferują swoim użytkownikom oraz usług sieci PI CAN, jakie w tym celu wykorzystują. Omówione zostały również usługi zewnętrzne, tj. Serwer Treści, Serwer Wyszukiwania Treści oraz Serwer AAA, w doniesieniu do każdej z aplikacji.




**Rysunek 2.** Komponenty aplikacji HomeNetMedia

### 3.1 HomeNetMedia

HomeNetMedia to aplikacja oferująca spersonalizowany dostęp do treści multimedialnych w sieci domowej, poprzez integrację treści pochodzących z różnych serwerów oraz integrację urządzeń odtwarzających. Potrzeba integracji podyktowana jest stale rosnącą liczbą oraz różnorodnością urządzeń multimedialnych, zarówno udostępniających treści, jak i je odtwarzających, dostępnych w sieci domowej. Przeciętnemu użytkownikowi coraz trudniej jest zapanować nad tą różnorodnością i w pełni wykorzystać dostępne możliwości. Co więcej, użytkownicy w coraz większym stopniu korzystają z zasobów dostępnych przez Internet, takich jak internetowe wypożyczalnie filmów, serwisy muzyczne czy usługi typu Internet TV. Zwiększają one w istotny sposób liczbę źródeł treści multimedialnych.

Repozytorium treści udostępnianych przez HomeNetMedia obejmuje zarówno zasoby lokalne, jak i treści zdalne pochodzące od różnych dostawców. Niezależnie od źródła pochodzenia, treści te mogą być w sposób transparentny dla użytkownika odtwarzane na wybranym urządzeniu w sieci domowej. Aby jednak efekt taki można było uzyskać, zarówno czas dostępu, jak i jakość przekazu dla treści zdalnych muszą być porównywalne z czasem i jakością uzyskiwaną w przypadku treści pobieranych z serwerów w sieci lokalnej. Spełnienie tego wymagania jest możliwe dzięki usługom sieci świadomej treści.

Integracja treści i urządzeń w sieci domowej możliwa jest dzięki wykorzystaniu technologii UPnP [4], w szczególności architektury UPnP AV [5]. Przyjęto za-

łożenie, ze urządzenia multimedialne w sieci domowej spełniają wymagania zdefiniowane przez DLNA Guidelines [6] określające m.in. zestaw formatów treści, które urządzenia odtwarzające muszą akceptować, a serwery mediów udostępniać dla danego rodzaju treści. Integracja treści dostępnych przez PI CAN z zasobami lokalnymi realizowana jest przez Home Gateway. Komponent ten współpracuje z siecią PI CAN wyszukując i pobierając treści w niej dostępne, w sieci domowej zaś realizuje funkcjonalność media serwera udostępniając te treści do odtwarzania na urządzenia lokalnych. Broker Treści pełniący funkcję punktu kontrolnego UPnP, tworząc zintegrowany katalog nie jest świadomy pochodzenia treści zgłaszanych przez Home Gateway. Podobnie w przypadku odtwarzania tych treści, Home Gateway pełni rolę tzw. proxy pośrednicząc w pobraniu treści przez PI CAN. Z punktu widzenia urządzeń odtwarzających, Home Gateway jest źródłem tych treści. Rys. 2 przedstawia komponenty aplikacji HomeNetMedia oraz powiązanie z siecią PI CAN.

HomeNetMedia realizuje również personalizację oraz dostosowanie usługi do kontekstu, w którym jest oferowana, tj. wybór treści oraz urządzenia odtwarzającego zależnie od preferencji użytkownika i kontekstu. W odniesieniu do wyboru urządzenia, kontekst zdefiniowany jest przez lokalizację użytkownika oraz wybraną treści. Przy czym drugi z tych elementów określa formaty jakie musi akceptować oraz protokół jaki musi realizować urządzenie odtwarzające.

Charakter treści oraz mechanizm udostępniania treści w sieci domowej wpływają w dużym stopniu na sposób realizacji usług zewnętrznych czyli CS, CSS oraz AAAS. Metadane treści udostępnianie są przez wszystkie serwery w sieci domowej w formacie DIDL Lite [7]. Zasady wyszukiwania treści określone są przez UPnP AV dla usługi Content Directory Service [4] implementowanej przez każdy serwer. Dla treści publikowanych w sieci PI CAN przyjęto taki sam format metadanych oraz zasady wyszukiwania. W związku z tym Serwer Wyszukiwania Treści musi interpretować zapytania zgodnie z wytycznymi UPnP AV w odniesieniu do wskazanych pól w metadanych opisowych opublikowanych treści. Typowe zlecenie wyszukania treści określa słowa występujące w tytule, nazwę wykonawcy, gatunek muzyki czy rodzaj filmu.

Serwer AAA dokonuje autoryzacji użytkowników w sposób specyficzny dla aplikacji. Zwykle autoryzacja dokonywana jest na podstawie danych pochodzących od dostawcy treści określających uprawnienia użytkownika do pobrania treści o wskazanym identyfikatorze. Dane te mogą odzwierciedlać np. wykupienie przez użytkownika abonamentu u określonego dostawcy, pozwalającego na określoną liczbę pobrań treści z określonej grupy.

### 3.2 Rozproszone Wirtualne Muzeum

Rozproszone Wirtualne Muzeum (ang. *Distributed Virtual Museum*, DVM) jest przykładem aplikacji klasy 3C (ang. *customize, create, contribut*e). W rozwinięciu oznacza możliwość tworzenia nowych i łączenia istniejących obiektów (create) dostosowanych do potrzeb odbiorców (customize) poprzez ich czynny udział w procesie kreacji (contribute), co stanowi fundamentalne założenia dla tzw. kreatywnej gospodarki (kreatywne przemysły, kreatywne sektory, itp.) [8].

Podstawowa funkcjonalność tej aplikacji bazuje na interaktywnym dostarczaniu treści 3D. Może być ona zastosowana w dwóch podstawowych scenariuszach:

1. przygotowanie i publikacja ekspozycji,
2. zwiedzanie ekspozycji.

W związku z tym można wyróżnić dwa rodzaje użytkowników: kurator wystawy i zwiedzający. Rolą kuratora jest przygotowanie niezależnych scen z eksponatami i połączenie ich w ekspozycję. Do budowy scen są wykorzystane opublikowane wcześniej w sieci obiekty 3D, których lokalizacja może być rozproszona. Podobnie do przygotowania ekspozycji mogą zostać wykorzystane istniejące sceny, których opisy też mogą mieć rozproszoną lokalizację. Powiązanie pomiędzy sceną a eksponatem może być zrealizowane poprzez unikatowy identyfikator lub poprzez opis cech takiego obiektu. Definiowanie obiektu poprzez opis jego cech w połączeniu z usługą wyszukiwania treści PI CAN pozwala na tworzenie scen, których wygląd będzie różny w zależności od dostępności obiektów w sieci. Przykładowo opis sceny może zawierać opis wymagający prezentacji rzeźby określonego autora wykonanej w zadanym okresie historycznym, jednak bez wskazania konkretnego dzieła.

Zwiedzający ma możliwość wyszukania interesującej go tematycznie ekspozycji, a następnie interaktywnego zwiedzania poszczególnych scen. Sceny i mieszczące się na nich obiekty są wyszukiwane i pobierane w zależności od akcji podejmowanych przez użytkownika. Ważną cechą aplikacji jest zastosowanie kodowania progresywnego do eksponatów o reprezentacji wektorowej. Pozwala ona na realizację progresywnego pobierania obiektów 3D w zależności od położenia zwiedzającego na scenie w relacji do oglądanych obiektów. Dla eksponatów znajdujących się bliżej i wymagających więcej szczegółów, pobieranych jest więcej danych niż dla obiektów odległych, co ma w założeniu wpłynąć na gospodarniejsze wykorzystanie zasobów sieci oraz stacji użytkownika [9].

Do realizacji takich scenariuszy zostały zdefiniowane cztery komponenty aplikacji:

1. Serwer Ekspozycji i Scen (ang. *Exhibition and Scen Server*, ESS) – służy do składowania i udostępniania opisów ekspozycji i scen wchodzących w ich skład,
2. Serwer Magazynowy (ang. *Store House Server*, SHS) – służy do składowania i udostępniania eksponatów wraz z informacjami uzupełniającymi,
3. Kurator Wirtualnej Wystawy (ang. *Virtual Exhibition Curator*, VEC) – jest przeznaczony do tworzenia opisów wystaw i scen (w tym wyszukiwania scen i obiektów) oraz do umieszczania ich, wraz z wymaganymi eksponatami na odpowiednich, wyżej wymienionych serwerach,
4. Inteligentny Eksplorator Wystawy (ang. *Smart Exhibition Explorer*, SEE) – jest oprogramowaniem klienckim pozwalającym użytkownikowi wyszukać interesującą go wystawę, a następnie zwiedzać składające się na nią sceny 3D wizualizowane z wykorzystaniem technologii stereoskopowych.

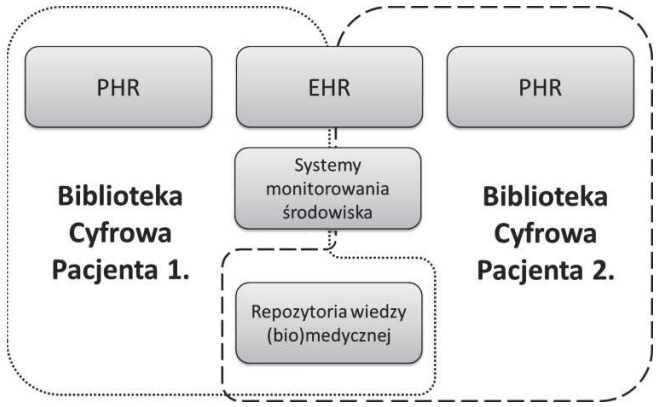
### 3.3 Biblioteka Cyfrowa Pacjenta

Biblioteka Cyfrowa Pacjenta jest aplikacją mającą na celu integrację oraz udostępnianie wszelkich dostępnych w sieci treści dotyczących zdrowia poszczególnych osób (bezpośrednich użytkowników aplikacji). Treści te przetwarzane są następnie w ramach szeregu różnorodnych scenariuszy z zakresu e-Zdrowia, umożliwiając swobodny dostęp do informacji niezbędnych w procesach diagnostyki, terapii, rehabilitacji oraz profilaktyki wszystkim aktorom w nie zaangażowanym, niezależnie od ich aktualnej lokalizacji.

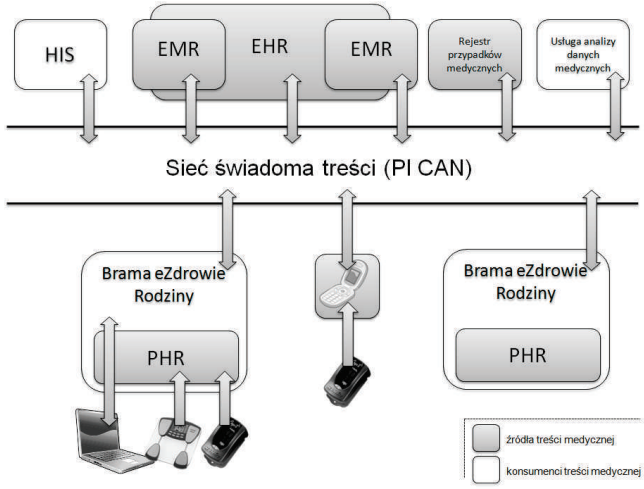
Biblioteka Cyfrowa Pacjenta jest zorientowanym na pacjenta narzędziem zarządzania danymi medycznymi jego dotyczącymi. Dane te obejmują przede wszystkim historie chorób i wyniki badań, zapisywane i przechowywane w ramach systemu ochrony zdrowia przez systemy informatyczne klasy EMR (ang. *Electronic Medical Record*), grupowane na poziomie regionalnym lub krajowym w platformy klasy EHR (ang. *Electronic Health Record*), informacje medyczne generowane przez pacjenta (np. w ramach aktywności fizycznej czy profilaktyki domowej) i przechowywane przez platformy klasy PHR (ang. *Personal Health Record*) oraz wiedzę (bio)medyczną przechowywaną w różnego rodzaju repozytoriach faktów medycznych, a także dane pochodzące z systemów monitorowania środowiska, w którym dany pacjent przebywa [10]. Powyższe dane przetwarzane są w indywidualnym kontekście każdego pacjenta, z możliwością dynamicznego włączania odpowiednich repozytoriów danych medycznych do spersonalizowanej rozproszonej biblioteki cyfrowych danych medycznych (Rys. 3).

Z uwagi na to, że podstawowym założeniem przyjętym przy budowie Biblioteki Cyfrowej Pacjenta było określenia pacjenta (właściciela treści) jako zarządcy informacji medycznej jego dotyczącej, niezwykle istotne stało się określenie sposobu realizacji platformy PHR stanowiącej element składowy środowiska Biblioteki. Zdecydowano o umieszczeniu spersonalizowanego repozytorium PHR w środowisku sieci domowej pacjenta, ułatwiając tym samym zarówno zarządzanie i zachowanie prywatności składowanych w nim danych, jak i realizację scenariuszy profilaktyki zdrowotnej z wykorzystaniem osobistych urządzeń medycznych. Repozytorium to zaimplementowano jako moduł tzw. Bramy eZdrowie Rodziny, stanowiącej węzeł pośredniczący w komunikacji i wymianie danych pomiędzy siecią domową pacjenta a systemami zewnętrznymi. Węzeł ten, obok funkcjonalności platformy PHR, implementuje także moduły akwizycji i wizualizacji danych medycznych w środowisku sieci domowej.

Brama eZdrowie Rodziny stanowi podstawowy serwer treści implementowany w ramach prototypu Biblioteki Cyfrowej Pacjenta opartej o sieć PI CAN. Prototyp obejmuje także serwer treści stanowiący symulację systemu klasy EHR. Założono ponadto dołączenie do prototypowej Biblioteki repozytorium faktów medycznych opartego o rejestr przypadków medycznych wykonany w ramach prac prowadzonych w projekcie pn. „Wielkopolskie Centrum Telemedycyny", urządzeń mobilnych umożliwiających akwizycję i dostęp do danych oraz demonstracyjnych usług analizy danych medycznych. Architekturę prototypu Biblioteki Cyfrowej Pacjenta przedstawiono na Rys 4.




**Rysunek 3.** Integracja danych medycznych w ramach spersonalizowanych Bibliotek Cyfrowych Pacjenta




**Rysunek 4.** Architektura prototypu Biblioteki Cyfrowej Pacjenta

Jednym z najważniejszych założeń przyjętych w implementacji Biblioteki Cyfrowej Pacjenta jest zastosowanie standardu wymiany danych medycznych w celu zapewnienia interoperacyjności systemów korzystających treści udostępnianych w sieci. Przyjęto zatem wykorzystanie powszechnie stosowanego w systemach klasy EHR standardu HL7 CDA (ang. *Health Level 7 Clinical Document Architecture*) [11] jako podstawowej struktury zapisu informacji medycznej w implementowanej aplikacji.

Z uwagi na specyficzne wymagania odnośnie interoperacyjności treści medycznych, przyjęto także założenie własnej implementacji Serwera Wyszukiwa-

nia Treści. Serwer ten zakłada wykorzystanie standardowych klasyfikacji do tworzenia opisu metadanowego treści medycznych. Zastosowane standardy to klasyfikacje ICD (ang. *International Statistical Classification of Diseases and Related Health Problems*) [12] w zakresie kodowania chorób i procedur medycznych oraz LOINC (ang. *Logical Observation Identifiers Names and Codes*) [13] w zakresie kodowania badań. Ponadto, metadane zawierają informacje dotyczące pacjenta, którego treść dotyczy, a także odniesienie do użytkownika sieci PI CAN, który opublikował daną treść w sieci (tj. pacjent, szpital, przychodnia).

Serwer Wyszukiwania Treści ściśle współpracuje z Serwerem AAA w zakresie autoryzacji dostępu do, w większości wrażliwych, treści medycznych. Podobnie jak dla Serwera Wyszukiwania Treści, tak i w tym wypadku Biblioteka Cyfrowa Pacjenta wprowadza własną implementację usługi sieci PI CAN. Serwer AAA w zastosowaniu dla e-Zdrowia autoryzuje dostęp do treści już na etapie przekazywania wyników wyszukiwania treści w sieci: treści, do których danych użytkownik nie posiada uprawnień, nie są umieszczanie na liście treści znalezionych w odpowiedzi na przekazane zapytanie. Autoryzowany jest także dostęp do właściwej treści medycznej. Uprawnienia, zgodnie z wymaganiem ustanowienia pacjenta jako zarządcy danych do niego należących, nadawane są przez niego.

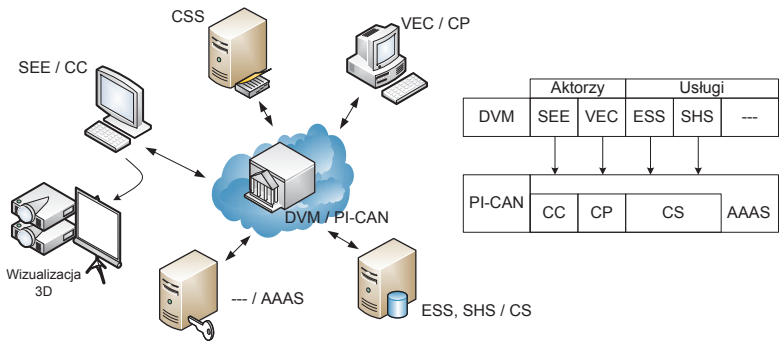
## 4 Integracja aplikacji z CAN

Wszystkie przedstawione w poprzednich rozdziałach aplikacje zostały zintegrowane z siecią PI CAN. Poniżej opisane zostały szczegóły dotyczące integracji dla każdej z aplikacji z uwzględnieniem jej specyfiki. Przedstawione zostały również przeprowadzone testy oraz ich wyniki.

### 4.1 HomeNetMedia

W celu integracji aplikacji HomeNetMedia z siecią świadomą treści, do PI CAN podłączony został Home Gateway jako element łączący sieć domową z usługami zewnętrznymi. Do jego konfiguracji wpisany został adres punktu dostępowego sieci PI CAN oraz dane potrzebne do uwierzytelnienia użytkownika zarejestrowanego w Serwerze AAA. Home Gateway podłączony został również do sieci domowej (lokalnej), do której podłączonych było kilka serwerów mediów oraz urządzeń odtwarzających treści muzyczne i treści wideo.

W ramach przygotowania do testów, w PI CAN opublikowanych zostało kilka utworów muzycznych oraz treści wideo, wraz z odpowiednim zestawem metadanych. Dla wybranego użytkownika zdefiniowany został profil z listą odtwarzania składającą się z utworów muzycznych pochodzących z lokalnego serwera mediów (Windows Media Serwer) oraz utworów opublikowanych w PI CAN. Lista ta definiowana była na podstawie danych pochodzących ze zintegrowanego katalogu treści tworzonego przez Broker Treści, a więc zgłaszanych odpowiednio przez lokalny serwer mediów oraz Home Gateway. Następnie utwory z listy odtworzone zostały na wybranym urządzeniu w sieci domowej. Nie wystąpiła zauważalna różnica w odtwarzaniu składowych listy pochodzących z różnych źródeł.



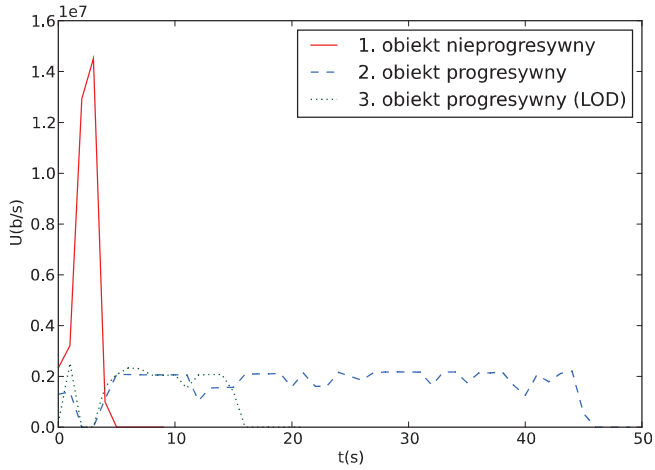

**Rysunek 5.** Odwzorowanie użytkowników i komponentów DVM na aktorów i usługi PI-CAN

Druga cześć testów dotyczyła treści wideo i polegała na odtworzeniu treści opublikowanej w sieci PI CAN na wybranym urządzeniu w sieci lokalnej. Odtwarzanie treści udostępnianej przez Home Gateway na lokalnym urządzeniu zlecone zostało przez standardowy kontroler UPnP, który dostępny jest na wielu urządzeniach w sieci domowej, w tym również mobilnych. Test ten miał na celu sprawdzenie funkcjonowania Home Gateway'a jako serwera mediów, dzięki któremu treści publikowane w PI CAN są dostępne bez konieczności używania specjalnego oprogramowania czy sprzętu. Odtwarzanie treści pochodzącej z PI CAN zlecone zostało również przez Brokera Treści jako kontrolera oferującego dodatkowe usługi związane z integracją treści i urządzeń oraz personalizacją usługi w ramach HomeNetMedia. W tym drugim przypadku Broker Treści dokonał wyboru urządzenia odtwarzającego na podstawie informacji o lokalizacji użytkownika.

### 4.2 Rozproszone Wirtualne Muzeum

Integracja polegała na odpowiednim odwzorowaniu komponentów aplikacji DVM na usługi PI CAN (Rys 5) oraz z umożliwieniem pobierania treści w sposób progresywny z SHS. Zmodyfikowane zostało działanie CS poprzez wprowadzenie do adresu obiektu dodatkowej informacji identyfikującej pożądany fragment treści. SHS informowany jest o rozpoczęciu pobierania progresywnego, co pozwala mu podtrzymać połączenie i w efektywny sposób dostarczać kolejne fragmenty obiektów w odpowiedzi na żądania od SEE. Połączenie tego typu może być również zakończone na żądanie SEE przed pobraniem całego obiektu (np. gdy użytkownik przechodzi do innej sceny a aktualnie oglądane obiekty nie będą już potrzebne).

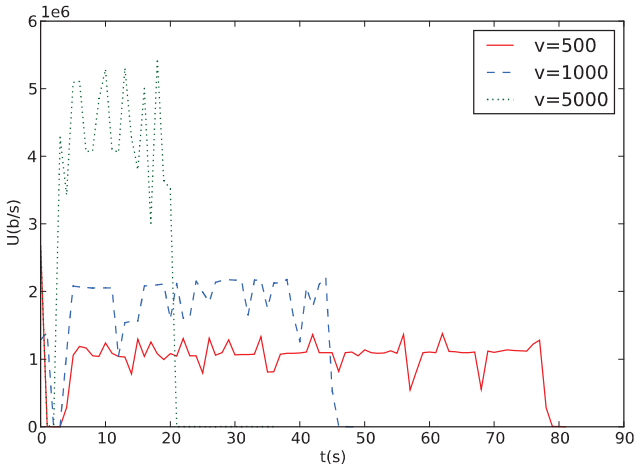
Wykorzystując mechanizmy pobierania i wyszukiwania treści w PI-CAN, obiekty zdefiniowane w poszczególnych ekspozycjach i scenach, mogą być identyfikowane na dwa sposoby: poprzez ich unikalny identyfikator lub poprzez związane z nimi meta znaczniki (definicja opisowa). Jeżeli obiekt zdefiniowany jest




**Rysunek 6.** Wykorzystanie pasma U(t) podczas transmisji nieprogresywnej i progresywnej

opisowo, przed jego pobraniem SEE używa funkcji wyszukiwania w celu lokalizacji pasujących zasobów (np. do personalizacji sceny lub zastępowania obiektów niedostępnych).

Podczas integracji przeprowadzono testy funkcjonalne aplikacji DVM oraz zweryfikowano zasadność stosowania transmisji progresywnych. Weryfikację przeprowadzono dla transmisji z jednym SHS i jednym SEE. Większość prezentowanych obiektów, pobierana jest progresywnie - początkowo na prezentowanej scenie przygotowywane są uproszczone wersje eksponatów (typowo o szczegółowości około 5% maksymalnej). Obiekty bliskie obserwatora i widoczne dla niego, są uzupełniane na bieżąco. Pozwala to uniknąć wstępnego ładowania znaczniej liczby dużych obiektów oraz zaoszczędzić dostępne zasoby sieciowe. Użytkownik zbliża się do obiektu aż do uzyskania satysfakcjonującego poziomu szczegółowości, pozwalającego na jego komfortową obserwację. Na Rys. 6 przedstawiono wynik przykładowego eksperymentu polegającego na porównaniu wykorzystania dostępnego pasma przy zastosowaniu transmisji progresywnej oraz bez niej. Na scenę ładowany był ten sam obiekt i użytkownik przemieszczał się na scenie w ten sam sposób. Widoczne jest, że transmisja nieprogresywna (krzywa 1) wykorzystuje w krótkim okresie czasu znaczącą ilość dostępnego pasma, ponieważ cały obiekt jest pobierany od razu po załadowaniu sceny. W przypadku transmisji progresywnej fragmenty obiektu są pobierane stopniowo, w związku z tym pasmo jest wykorzystywane przez dłuższy czas jednak w znacznie mniejszym stopniu (krzywa 2), ponadto pobieranie progresywne może zostać przerwane (krzywa 3), jeśli jakość prezentowanego obiektu będzie zadowalająca dla użytkownika. Niewielkie opóźnienie na początku, widoczne dla scenariuszy progresywnych, związane jest z implementacją ładowania obiektów na scenę. Obiekty nieprogre-




**Rysunek 7.** Wykorzystanie pasma U(t) podczas transmisji progresywnej dla różnych kroków aktualizacji

sywne muszą zostać pobrane w całości i rozpakowane przed jej utworzeniem, co wymaga większych zasobów pamięciowych od SEE.

Płynna prezentacja treści progresywnych wymaga określenia dwóch istotnych parametrów: funkcji doboru poziomu szczegółowości w zależności od odległości obserwatora od obiektu Flod oraz liczby wierzchołków pobieranych z SHS w czasie jednego żądania (krok aktualizacji v). Wykorzystanie pasma dla różnych wartości parametru v przy zastosowaniu liniowej funkcji Flod przedstawiono na Rys. 7. Część dalszych prac związanych z DVM będzie dotyczyła między innymi opracowania metody zautomatyzowania doboru wartości tych parametrów.

### 4.3 Biblioteka Cyfrowa Pacjenta

Integracja zaimplementowanego prototypu Biblioteki Cyfrowej Pacjenta ze środowiskiem PI CAN związana była w pierwszej kolejności z konfiguracją i podłączeniem do sieci świadomej treści poszczególnych usług implementowanych w ramach aplikacji. Do sieci PI CAN podłączono trzy serwery treści: dwa serwery oparte o Bramę eZdrowie Rodziny z modułami PHR oraz serwer treści odpowiedzialny za udostępnianie przykładowych treści medycznych powstających w zorganizowanej służbie zdrowia. Na serwerach zamieszczono: zestaw kilkunastu pomiarów wykonanych osobistymi urządzeniami medycznymi (pulsoksymetr oraz waga cyfrowa z analizatorem składu ciała) w repozytoriach PHR oraz kilka różnorodnych dokumentów HL7 CDA, w tym dokumentów zawierających w swojej strukturze badania obrazowe zapisane w standardzie DICOM 3.0. Warto zwrócić uwagę, że w odniesieniu do udostępnianych przez moduły PHR pomiarów dokonanych osobistymi urządzeniami medycznymi rejestrowanym w sieci PI CAN plikiem jest zbiór wszystkich pomiarów danego typu dla danego pacjenta (np.

historia pomiarów wagi ciała). Przyjęcie takiego rozwiązania jest bardziej efektywne z punktu widzenia sieci PI CAN oraz z punktu widzenia aplikacji wykorzystującej treści niż rejestrowanie każdego pomiaru jako osobnego pliku. Obok wymienionych wyżej serwerów treści, w sieci PI CAN skonfigurowano specyficzne dla Biblioteki Cyfrowej Pacjenta Serwer Wyszukiwania Treści oraz Serwer AAA.

W tak przygotowanym środowisku prototypu Biblioteki Cyfrowej Pacjenta przeprowadzono kilka testów weryfikujących działanie sieci świadomej treści dla aplikacji z obszaru eZdrowia. Pierwszy test polegał na uruchomieniu poszczególnych serwerów treści (modułów PHR na Bramach eZdrowie Rodziny oraz serwera treści symulującego system klasy EHR) w środowisku sieci PI CAN. Po uruchomieniu każdy z serwerów treści rejestrował w sieci (publikował) przechowywane przez siebie treści medyczne. Następnie za pomocą Bram eZdrowie Rodziny i skonfigurowanych wcześniej użytkowników aplikacji dokonano odczytu z PI CAN dostępnych dla nich treści. Bramy, poprzez komunikację z brzegowym węzłem PI CAN, kontaktowały się z Serwerem AAA, uwierzytelniając użytkownika, oraz Serwerem Wyszukiwania Treści, otrzymując listę treści dostępnych w sieci dla danego użytkownika. Listy te wyświetlane były w odpowiedni sposób w interfejsie użytkownika. Dokonywano następnie pobrania wybranego dokumentu w celu prezentacji ich na ekranie użytkownika.

Drugim testem przeprowadzonym w ramach integracji prototypu aplikacji z PI CAN było przeprowadzenie akwizycji nowych pomiarów wykonanych osobistymi urządzeniami medycznymi i umieszczenie ich w repozytorium PHR z wykorzystaniem mechanizmów sieci świadomej treści. W tym celu posłużono się pulsoksymetrem oraz urządzeniem mobilnym (telefon komórkowy typu smartphone) podłączonym poprzez wpięty do węzła PI CAN router sieci bezprzewodowej WLAN. Komunikacja pomiędzy pulsoksymetrem a telefonem realizowana była z wykorzystaniem protokołu Bluetooth. Test polegał na wykonaniu pomiaru, którego wynik następnie był, po wcześniejszym zatwierdzeniu przez użytkownika, transmitowany do odpowiedniej Bramy eZdrowie wskazanej adresem skonfigurowanym w mobilnym interfejsie użytkownika aplikacji. Przeprowadzenie takiej operacji powodowało modyfikację treści związanych z historią wskazań poziomu saturacji krwi oraz pulsu danego użytkownika. Zmodyfikowana treść była następnie odczytywana z odpowiedniego serwera treści i wyświetlana na ekranie użytkownika.

Opisane powyżej testy wykazały, że sieć świadoma treści może z powodzeniem stanowić środowisko realizacji scenariuszy integracji oraz współdzielenia danych medycznych, w szczególności w zakresie interakcji pomiędzy systemami klasy PHR oraz EHR/EMR.

## 5 Podsumowanie

W artykule przedstawiono trzy aplikacje korzystające z sieci świadomej treści PI CAN opracowanej w ramach projektu Inżynieria Internetu Przyszłości. Aplikacje te oferują użytkownikom dostęp do treści multimedialnych w różnych celach, jednakże ich wspólnym elementem są wymagania stawiane sieci, przez którą po-

bierane są treści. Dla każdej z aplikacji przedstawiona została funkcjonalność oferowana użytkownikom oraz sposób wykorzystania sieci świadomej treści. Ponadto, omówiona została funkcjonalność oraz implementacja usług zewnętrznych wymaganych do współpracy z siecią PI CAN, a specyficznych dla aplikacji. Przedstawiony został sposób integracji poszczególnych aplikacji z siecią PI CAN oraz wyniki przeprowadzonych testów. Testy te przebiegły pomyślnie pozwalając zademonstrować funkcjonalność aplikacji, których realizacje potwierdziły zasadność koncepcji sieci świadomej treści.

## Literatura


1. A. Dobrev, T. Jones, K. Stroetmann, Y. Watter, K. Peng, „The socio-economic impact of interoperable electronic health record (EHR) and ePrescribing systems in Europe and beyond", 2009
2. W. Burakowski, H. Tarasiuk, A. Bęben, „Architektura systemu IIP", Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź 2011
3. A. Bęben, et al., „Architektura sieci świadomych treści w systemie IIP", Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Łódź 2011
4. UPnP Specifications, [Online] http://upnp.org/sdcps-and-certification/standards/
5. J. Ritchie, T. Kuehnel, J. Kang, W. van der Beek, "UPnP AV Architecture:1", Standardized DCP, September 30, 2008
6. The Digital Living Network Alliance, "DLNA Networked Device Interoperability Guidelines", March 2006
7. UPnP Forum, "XML Schema for ContentDirectory Structure and Metadata (DIDL-Lite)", December 31, 2010
8. „Analiza potrzeb i rozwoju przemysłów kreatywnych w Polsce", Ekspertyza zamieszczona na stronie internetowej Ministerstwa Gospodarki, 2010, http://www.mg.gov.pl/files/upload/10147/ Analiza potrzeb i rozwoju przemyslow kreatywnych.pdf
9. K. Skabek, W. Winiarczyk, A. Sochan , "Implementation of the Progressive Meshes for Distributed Virtual Museum", Advances in Intelligent and Soft Computing 2012, Volume 98, Human – Computer Systems Interaction: Backgrounds and Applications 2
10. M. Kosiedowski, C. Mazurek, A. Stroiński, „Integration and sharing of personal health data as a challenge for the Future Internet e-Health scenarios", IADIS International Conference e-Health 2011, Rzym, str. 53-60
11. R. H. Dolin et al., „The HL7 Clinical Document Architecture", Journal of American Medical Informatics Association vol. 8 issue 6, 2001, str. 552-569
12. G. R. Brämmer, „International statistical classification of diseases and related health problems. Tenth revision.", World Health Statistics Quarterly vol. 41 issue 1, 1988, str. 32-36
13. A. W. Forrey et al., „Logical observation identifier names and codes (LOINC) database: a public use set of codes and names for electronic reporting of clinical laboratory test results", Clinical Chemistry vol. 42 issue 1, 1996, str. 82-90

# Portal PL-LAB

Łukasz Dolata[1], Radosław Krzywania[1], Miłosz Przywecki[1],
Piotr Krawiec[2]

[1] Poznańskie Centrum Superkomputerowo-Sieciowe
[2] Instytut Łączności – Państwowy Instytut Badawczy

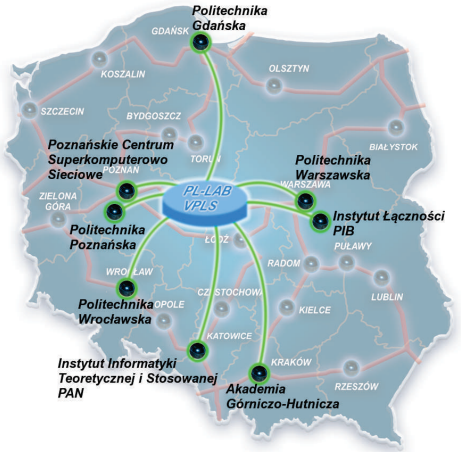
**Streszczenie** PL-LAB to rozproszone laboratorium umożliwiające przeprowadzanie eksperymentów dotyczących różnorodnych zagadnień związanych z badaniami w obszarze Internetu Przyszłości. Artykuł prezentuje portal PL-LAB, który zapewnia zdalny dostęp do infrastruktury i usług oferowanych przez sieć eksperymentalną PL-LAB. Portal ten pozwala użytkownikom na zdefiniowanie topologii sieci, w której chcą przeprowadzić eksperyment, oraz rezerwację koniecznych zasobów. Z kolei administratorzy systemu przy pomocy portalu mogą w sposób automatyczny zarządzać siecią PL-LAB.

## 1 Wprowadzenie

Krajowa sieć eksperymentalna PL-LAB [1,2] stworzona została w ramach projektu Inżynieria Internetu Przyszłości (IIP) [3] w celu umożliwienia prowadzenia badań dotyczących przyszłych generacji sieci Internet. PL-LAB pozwala użytkownikom zestawiać własne wirtualne sieci testowe, realizowane w oparciu o urządzenia eksperymentalne będące na wyposażeniu laboratoriów poszczególnych partnerów projektu IIP. Laboratoria te (zwane węzłami PL-LAB), umiejscowione są następujących ośrodkach akademickich i naukowo-badawczych (rys. 1):

- Akademii Górniczo-Hutniczej (AGH),
- Instytucie Informatyki Teoretycznej i Stosowanej Polskiej Akademii Nauk (IITiS),
- Instytucie Łączności - Państwowym Instytucie Badawczym (IŁ),
- Poznańskim Centrum Superkomputerowo-Sieciowym (PCSS),
- Politechnice Gdańskiej (PG),
- Politechnice Poznańskiej (PP),
- Politechnice Warszawskiej (PW),
- Politechnice Wrocławskiej (PWr).

Węzły PL-LAB połączone są ze sobą poprzez ogólnopolską sieć naukowo-badawczą PIONIER [4], w której uruchomiona została dedykowana usługa VPLS (ang. *Virtual Private LAN Service*). W ten sposób wszystkie laboratoria znalazły się wewnątrz jednej Ethernet'owej domeny rozgłoszeniowej. Połączenia sieciowe pomiędzy urządzeniami eksperymentalnymi zlokalizowanymi w różnych laboratoriach tworzone są za pomocą technologii VLAN [5] z gwarantowaną przepływnością. Zbiór urządzeń wybranych przez użytkownika spośród wszystkich udostępnionych w infrastrukturze PL-LAB wraz ze skonfigurowanymi między nimi




**Rysunek 1.** Umiejscowienie węzłów PL-LAB

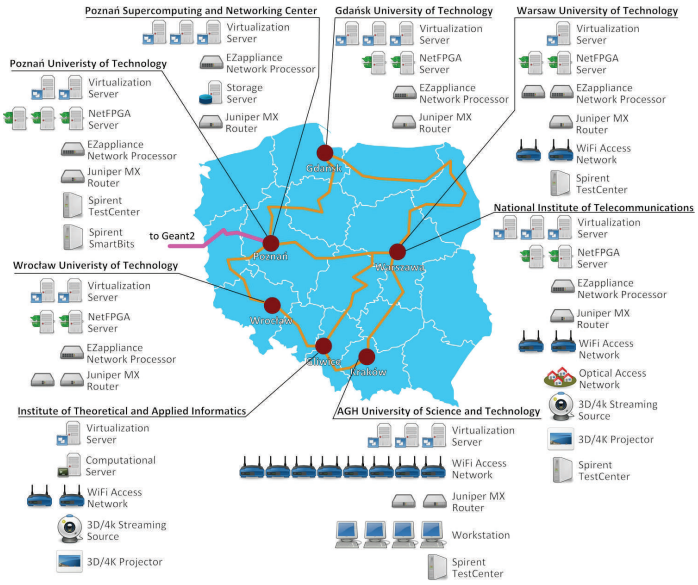
połączeniami sieciowymi tworzy dedykowane dla danego użytkownika wirtualne laboratorium, zwane Laboratorium Użytkownika, odizolowane nie tylko od ruchu przesyłanego w sieci PIONIER, ale również od innych przeprowadzanych w tym samym czasie w sieci PL-LAB eksperymentów.

Połączenia pomiędzy węzłami PL-LAB realizowane są za pomocą występujących w każdym węźle przełączników dostępowych Juniper EX 3200, których interfejs *uplink* podłączony jest do sieci PIONIER, zaś do pozostałych portów dołączone są urządzenia eksperymentalne. W przypadku węzłów PL-LAB umiejscowionych w ośrodkach PCSS oraz PG, przełącznik dostępowy podłączony jest do sieci PIONIER łączem o przepustowości 10 Gbit/s, pozostałe węzły wykorzystują natomiast łącza o przepustowości 1 Gbit/s. Należy jednak nadmienić, iż proces wymiany interfejsów z 1 GE na 10 GE jest w toku i w najbliższym czasie kolejne ośrodki zwiększą przepustowość łączy do sieci PIONIER do 10 Gbit/s.

Dostęp do infrastruktury PL-LAB zarówno dla użytkowników, jak i administratorów sieci, zapewniony jest poprzez opracowany w ramach projektu IIP System Zdalnego Dostępu [6]. System ten, dzięki posiadanym funkcjom zarządzania i planowania, umożliwia tworzenie wirtualnych Laboratoriów Użytkownika w sposób automatyczny, jak również izoluje całą sieć przed niepowołanym dostępem z zewnątrz. Interfejsem zapewniającym interakcję użytkowników z Systemem Zdalnego Dostępu jest Portal PL-LAB, którego funkcjonalność została przedstawiona w kolejnych rozdziałach niniejszego artykułu.

## 2 Laboratorium Użytkownika PL-LAB

PL-LAB stworzony został w ramach projektu IIP, którego zakres obejmuje opracowanie i przetestowanie prototypu nowej, innowacyjnej architektury sieciowej




**Rysunek 2.** Urządzenia wykorzystywane w węzłach sieci PL-LAB

budowanej w oparciu o techniki wirtualizacji, jak również korzystających z niej nowych typów usług i aplikacji. Powyższe cele projektu implikują następujące wymagania, jakie zostały nałożone na infrastrukturę tworzonej sieci badawczej:

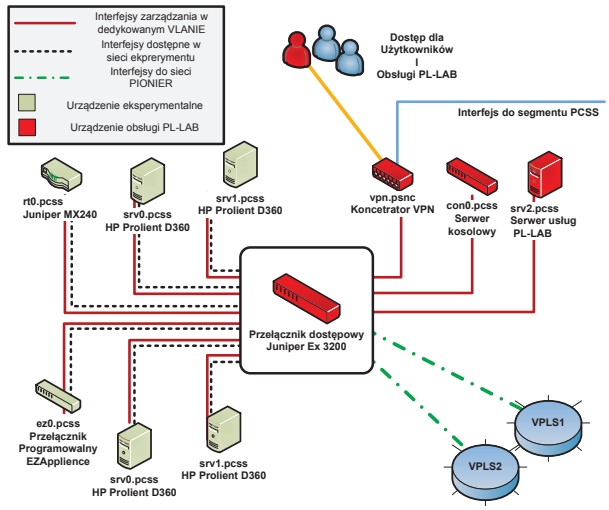
- infrastruktura powinna umożliwiać przeprowadzanie eksperymentów w jak najszerszym zakresie badawczym (protokoły sieciowe, mechanizmy wirtualizacji, aplikacje);
- infrastruktura powinna umożliwiać przeprowadzanie eksperymentów nowych rozwiązań, niekompatybilnych z obecnie stosowanymi technologiami opartymi na protokole IP;
- infrastruktura powinna umożliwiać niskopoziomowy (tj. do warstwy sprzętowej) dostęp do wykorzystywanych w eksperymentach urządzeń.

Przy uwzględnieniu powyższych wymagań zbudowane zostało rozproszone laboratorium PL-LAB, w ramach którego użytkownicy mogą tworzyć dedykowane, wirtualne Laboratoria Użytkownika. Laboratorium takie może być zbudowane w oparciu o następujące urządzenia udostępniane w ramach PL-LAB (rys. 2):

- programowalne urządzenia sieciowe: platformy z programowalnym procesorem sieciowym NP-3 EZappliance firmy EZchip oraz serwery z kartami NetFPGA (wyposażonymi w procesory Virtex2 lub Virtex5);
- rutery Juniper serii MX wspierające wirtualizację dla procesów sterowania oraz przekazu danych;
- serwery HP serii ProLiant z mechanizmami wsparcia wirtualizacji;

- urządzenia bezprzewodowych i optycznych sieci dostępowych;
- narzędzia pomiarowe Spirent TestCenter, umożliwiające sprzętową generację/pomiar ruchu sieciowego;
- urządzenia końcowe dla usług multimedialnych - kamery, kodery, dekodery oraz wyświetlacze dla wideokonferencji HD/3D oraz wideo wysokiej rozdzielczości 4K;
- telefony VoIP.

Dla zapewnienia niskopoziomowego dostępu do urządzeń tworzących dane Laboratorium użytkownicy mają możliwość komunikacji z urządzeniami już na poziomie warstwy drugiej (Ethernet) dzięki zastosowaniu mechanizmów VPN (ang. *Virtual Private Network*). Dostęp ten jest realizowany za pomocą wydzielonej sieci zarządzania (zbudowanej jako dedykowana sieć VLAN), do której podłączone są interfejsy dostępowe i konfiguracji urządzeń. Dołączenie użytkownika do sieci zarządzania odbywa się poprzez koncentratory VPN umiejscowione w każdym z węzłów PL-LAB. Dzięki temu użytkownik z dowolnego miejsca ma możliwość przeprowadzenia czynności koniecznych do przygotowania urządzeń tworzących jego Laboratorium Użytkownika dla wykonania eksperymentu: wgranie i instalacja wymaganego oprogramowania, wprowadzenie odpowiedniej konfiguracji itp.




**Rysunek 3.** Przykładowy schemat węzła PL-LAB

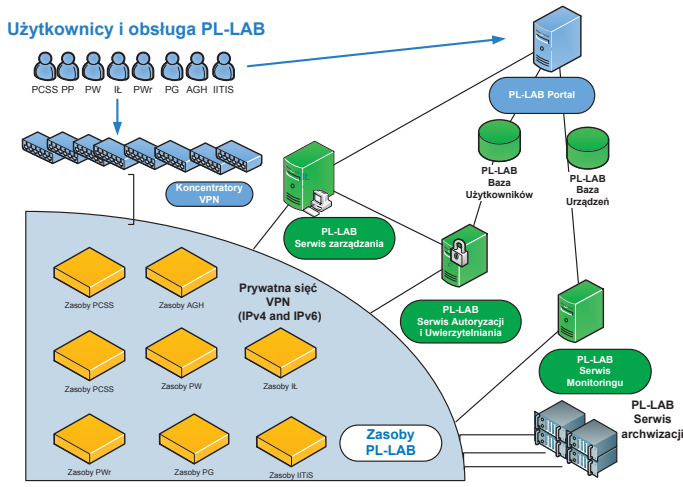
Każdy z węzłów został wyposażony w przełącznik Juniper EX3200/4200 zwany dalej przełącznikiem dostępowym (rys. 3). Interfejsy urządzeń eksperymentalnych, które są wpięte do przełącznika dostępowego, wykorzystywane są

w celu tworzenia połączeń wewnętrznych Laboratorium Użytkownika za pomocą VLAN-ów. Dane Laboratorium Użytkownika może zawierać dowolną liczbę połączeń zarówno w konfiguracji punkt - punkt, jak i punkt - wiele punktów. W celu jak najlepszego wykorzystania zasobów sieci PL-LAB oraz zapewnienia gwarancji dostępności zasobów dla przeprowadzanych równolegle eksperymentów, przepustowość połączeń dedykowanych dla danego Laboratorium Użytkownika limitowana jest ściśle do zapotrzebowania zgłoszonego przez użytkownika i ograniczana fizycznie na przełączniku dostępowym.

## 3 PL-LAB System zarządzania

Potrzeba opracowania rozbudowanych procedur oraz narzędzi zarządzania infrastrukturą PL-LAB została wymuszona specyfiką nowego środowiska badawczego, skupiającego aż osiem ośrodków w różnych miejscach Polski, zawierającego znaczną liczbę różnorodnego sprzętu, nie tylko sieciowego, oraz udostępniającego sieciowe laboratoria badawcze jednocześnie wielu użytkownikom. Prace nad stworzeniem odpowiednich narzędzi rozpoczęły się równolegle z procedurami zakupu i instalacji wyposażenia poszczególnych laboratoriów. We wczesnej fazie projektu zebrano niezbędne wymagania zarówno od użytkowników laboratorium jak i członków zespołu obsługi technicznej, z których najważniejsze to:

1. **Wsparcie dla obsługi technicznej PL-LAB w zarządzania zasobami i konfiguracji infrastruktury** - system powinien umożliwiać przechowywanie i zarzadzanie informacjami związanymi z infrastrukturą PL-LAB w tym: konfiguracja węzłów PL-LAB, konfiguracja oraz dostępność urządzeń zainstalowanych w infrastrukturze, konfiguracja Laboratoriów Użytkowników. Powinien również dostarczać narzędzia automatyzujące proces konfiguracji sprzętu jak również monitorować dostępność usług i zgłaszać wystąpienie awarii.
2. **Wsparcie użytkowników PL-LAB w procesie aplikowania o zasoby** - każdemu zarejestrowanemu użytkownikowi PL-LAB chcącemu przeprowadzić badania system powinien udostępnić zbiór narzędzi umożliwiających:
   - przegląd dostępnych w infrastrukturze PL-LAB zasobów oferując w ten sposób pomoc przy lokalizacji wymaganych urządzeń eksperymentalnych;
   - przygotowanie specyfikacji eksperymentu zawierającą listę wymaganego sprzętu, topologię połączeń sieciowych oraz wymagany poziom ich przepustowości;
   - konfigurację środowiska badawczego w oparciu o dostarczoną specyfikację bez konieczności angażowania użytkownika w ten proces;
   - dostęp i prowadzenie badań w nowo utworzonym środowisku eksperymentalnym;
   - zgłaszanie problemów z działaniem i użytkowaniem sieci eksperymentalnej.




**Rysunek 4.** Architektura Systemu Zdalnego Dostępu do PL-LAB

Uwzględniając powyższe założenia przygotowano specyfikację narzędzi, która stała się bazą prac implementacyjnych. Rezultatem prowadzonych prac jest system zarządzania nazwany Systemem Zdalnego Dostępu do sieci PL-LAB. System składa się z szeregu modułów funkcjonalnych a zależności miedzy nimi przedstawiono na rys. 4. W dalszej części rozdziału przedstawiono najważniejsze informacje dotyczące funkcjonalności każdego z modułów. Techniczne szczegóły realizacji zostały pominięte w opisie, gdyż wybiegają one poza ramy niniejszego artykułu. Pominięto również opis portalu PL-LAB, który w szczegółach zostanie przedstawiony w dalszych rozdziałach artykułu.

### PL-LAB Serwis Zarządzania

Moduł PL-LAB Serwis Zarządzania jest odpowiedzialny za wsparcie obsługi technicznej PL-LAB w procesie zarządzania, konfiguracji i utrzymania sprzętu. Umożliwia on przechowywanie szablonowych plików konfiguracyjnych oraz skryptów, wykorzystywanych do automatycznej i pół-automatycznej konfiguracji przełączników dostępowych oraz urządzeń eksperymentalnych.

### PL-LAB Serwis Monitorowania

Moduł PL-LAB Serwis Monitorowania odpowiedzialny jest za ciągłe monitorowanie stanu infrastruktury PL-LAB, poprzez zbieranie i udostępnianie informacji o aktualnej dostępności urządzeń PL-LAB oraz stopnia wykorzystania pasma na przełącznikach dostępowych. W przypadku wystąpienia problemów z działaniem sieci serwis powiadamia obsługę techniczną umożliwiając szybką reakcje i rozwiązywania problemów.

**PL-LAB Serwis Autoryzacji i Uwierzytelniania**
Moduł PL-LAB Serwis Autoryzacji i Uwierzytelniania dostarcza narzędzi umożliwiających autoryzację i uwierzytelnianie użytkowników oraz obsługi technicznej PL-LAB na sprzęcie zainstalowanym w infrastrukturze PL-LAB. Na potrzeby tego modułu zaadoptowano mechanizmy serwera RADIUS oraz serwera LDAP.

**Moduł obsługi PL-LAB VPN**
Moduł usługi VPN zapobiega nieautoryzowanemu dostępowi do infrastruktury PL-LAB. W ramach Systemu Zdalnego Dostępu PL-LAB stworzono prywatną sieć VPN połączoną z siecią zarządzania infrastruktury PL-LAB. Dostęp do ww. sieci VPN jest ograniczony wyłącznie do zarejestrowanych w systemie dostępowym PL-LAB użytkowników i może być uzyskany poprzez jeden z koncentratorów VPN zainstalowanych w każdym z węzłów sieci PL-LAB.

## 4 PL-LAB Portal

Sieć badawcza PL-LAB miała możliwość korzystania z doświadczeń i rozwiązań wypracowanych przez międzynarodowe projekty naukowe takie jak FEDERICA [7], OneLab [8], PlanetLab [9], których działania polegały na udostępnianiu infrastruktur sieciowych i obliczeniowych. Były one dodatkowym źródłem wiedzy i pomysłów, zarówno w aspektach architektonicznych, wymagań systemowych, zarządzania jak również obsługi użytkowników. Rezultaty tych projektów ułatwiły prace nad specyfikacją narzędzi dla sieci PL-LAB i pomogły określić wymagania, które powinny zostać spełnione przy tworzeniu infrastruktury PL-LAB.

Jedną ze wspólnych cech wyżej wymienionych projektów było udostępnianie użytkownikom portali dostępowych umożliwiających interakcję z danym środowiskiem badawczym, poprzez wsparcie dla procesu aplikowania i użytkowania wirtualnych środowisk eksperymentalnych. Dla zapewnienia jak największej dostępności, aplikacje te były tworzone jako publicznie dostępne serwisy internetowe. Zgodnie z przyjętą polityką bezpieczeństwa, możliwość korzystania z pełnej funkcjonalności portali mają wyłącznie zalogowani użytkownicy, po uprzedniej rejestracji w systemie. Tworzenie portali dostępowych ma wiele zalet, w tym między innymi usprawnia oraz ułatwia proces interakcji z infrastrukturą badawczą, ujednolica specyfikację zgłaszanych eksperymentów jak również umożliwia analizę wykorzystanych zasobów i lepsze planowanie badań użytkowników. Jednocześnie zapewnienie użytkownikom jednego, wspólnego punktu kontaktowego i narzędzia zarządzania upraszcza znacznie interakcję ze strony użytkownika.

Podobnie jak w przypadku wyżej wymienionych projektów, również przy tworzeniu Systemu Zdalnego Dostępu do sieci PL-LAB zdecydowano się na utworzenie dedykowanego serwisu internetowego. Początkowo planowano ograniczyć jego funkcjonalność wyłącznie do obsługi zgłoszeń eksperymentów użytkowników PL-LAB, jednakże ostatecznie dodano moduły zarządzania i automatycznej konfiguracji infrastruktury. Tak stworzony serwis, nazwany Portalem

PL-LAB, dostarcza niezbędne narzędzia umożlwiające interakcję użytkowników i obsługi technicznej z systemem PL-LAB. Przy użyciu serwisu użytkownicy PL-LAB mogą przygotowywać i zgłaszać w systemie PL-LAB specyfikacje własnych eksperymentów, nadzorować stan sieci badawczej oraz eksperymentów, a także uzyskać niezbędne informacje jak użytkować daną sieć badawczą. W przypadku obsługi technicznej PL-LAB, serwis umożliwia zarządzania informacją o stanie infrastruktury, w tym informacją o: konfiguracji węzłów, konfiguracji urządzeń, konfiguracji dostarczonych przez sieć PIONIER usług VPLS i konfiguracji VLAN-ów. Dostarcza również narzędzia umożliwiające automatyzację procesów konfiguracji, zarządzania i nadzoru infrastruktury PL-LAB.

Portal PL-LAB dostępny jest pod adresem `http://pl-lab.iip.net.pl` i w chwili obecnej umożliwia rejestrację wyłącznie uczestnikom projektu IIP. Po zakończeniu projektu zasoby sieci PL-LAB zostaną udostępnione również użytkownikom spoza projektu, jednakże wymaga to wprowadzenia niezbędnych zmian w polityce bezpieczeństwa i organizacji laboratoriów.

### 4.1 Struktura organizacyjna i uprawnienia

Jak wspominano wcześniej, dostęp do pełnej funkcjonalności portalu możliwy jest wyłącznie po zalogowaniu w systemie. W celu klasyfikacji użytkowników i odpowiadających im uprawnień zdefiniowano pięć ról funkcyjnych:

- Użytkownika (najniższy poziom uprawnień, osoba uprawniona do wykonywania eksperymentów w PL-LAB);
- Opiekuna węzła (osoba odpowiedzialna za węzeł PL-LAB i spójność informacji o konfiguracji węzła w portalu);
- Opiekuna eksperymentu (osoba odpowiedzialna za konfigurację i nadzór nad eksperymentami użytkowników);
- Opiekuna użytkowników (osoba odpowiedzialna za nadzór kont użytkowników danego węzła);
- Administratora (najwyższy poziom uprawnień).

Dana osoba korzystająca z portalu może mieć jednocześnie przypisanych kilka ról.

Rejestracja użytkowników odbywa się indywidulanie. Każdy z użytkowników przy rejestracji wskazuje węzeł PL-LAB, do którego przynależy. Konto użytkownika jest zablokowane do czasu akceptacji zgłoszenia przez Opiekuna użytkowników danego węzła, który potwierdza tożsamość użytkownika. Nadawanie użytkownikowi dodatkowych ról funkcyjnych następuje podczas procesu akceptacji i jest wykonywane przez Opiekuna użytkowników (dla ról: Opiekun węzła, Opiekun rezerwacji) lub Administratora (dla wszystkich ról).

W celu elastycznej obsługi różnych ról określających uprawnienia, serwis został podzielony na perspektywy funkcjonalne. Dostęp do każdej z nich limitowany jest poziomem odpowiednich uprawnień dla użytkownika. Na rys. 5 widoczne są zakładki, które symbolizują i aktywują dostępne perspektywy w serwisie. Każda aktywna perspektywa zmienia zestaw dostępnych funkcji wyszczególnionych w lewej części okna serwisu.

**Rysunek 5.** Perspektywy funkcjonalne portalu PL-LAB

Poniżej zestawiono opis dostępnych perspektyw wraz z wymaganym poziomem uprawnień:

- **PL-LAB Status** - dostęp dla wszystkich zalogowanych użytkowników - perspektywa umożliwia przegląd stanu infrastruktury PL-LAB, w tym informacji o węzłach wchodzących w skład infrastruktury, dostępnych urządzeniach eksperymentalnych, urządzeniach utrzymania systemu PL-LAB czy informacji o wszystkich prowadzonych eksperymentach.
- **Użytkownik PL-LAB** - dostęp dla wszystkich zalogowanych użytkowników - perspektywa dostarcza funkcje ułatwiające proces zgłoszenia eksperymentu, śledzenia procesu jego konfiguracji, użytkowanie infrastruktury eksperymentalnej oraz sygnalizowanie problemów z jej użytkowaniem.
- **Opiekun węzła PL-LAB** - dostęp limitowany dla użytkowników posiadających uprawnienia opiekuna węzła - perspektywa dostarcza funkcji ułatwiających zbieranie i zarządzanie informacjami o konfiguracji węzła sieci PL-LAB, użytkownikach węzła, obsłudze technicznej, konfiguracji elementów dostępowych do sieci PL-LAB VPN.
- **Opiekun eksperymentu** - dostęp limitowany dla użytkowników posiadających uprawnienia opiekuna eksperymentów - perspektywa dostarcza funkcji ułatwiających proces konfiguracji sieci eksperymentalnych zgłaszanych przez użytkowników PL-LAB.
- **Opiekun użytkowników PL-LAB** - dostęp limitowany dla użytkowników posiadających uprawnienia opiekuna użytkowników węzła - perspektywa dostarcza funkcje ułatwiające zarządzanie kontami użytkowników przynależących do węzła PL-LAB, w tym zatwierdzanie rejestracji, blokowanie istniejących kont oraz nadawanie/redukcja uprawnień.
- **Administrator PL-LAB** - dostęp limitowany dla użytkowników o najwyższym poziomie uprawnień - perspektywa dostarcza funkcje ułatwiające zarządzanie informacjami o wszystkich węzłach PL-LAB, urządzeniach zainstalowanych w sieci, zarządzanych pulach VLAN, dostępnych sieciach VPLS, wszystkich kontach użytkowników itp.

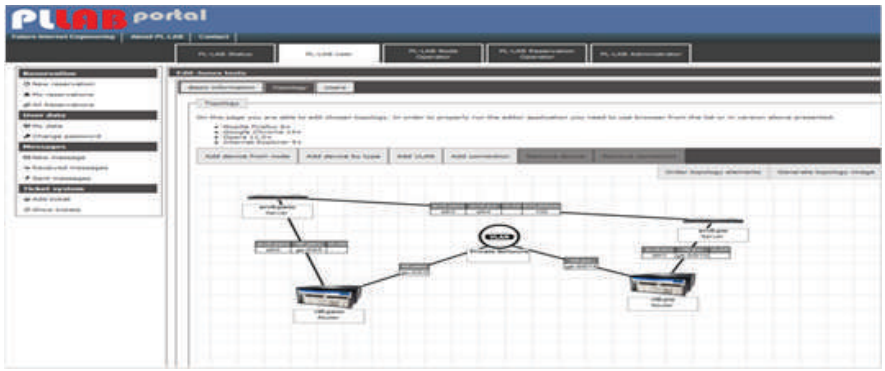
### 4.2 Wparcie dla użytkowników i ich eksperymentów

Sukces sieci badawczej PL-LAB w dużej mierze zależy od zadowolenia użytkowników korzystających w swoich badaniach z jej zasobów. Wpływ na to ma sposób w jaki użytkownik może aplikować, użytkować oraz monitorować status swojego wirtualnego środowisko badawczego. Niezbędne jest zatem wsparcie po stronie

narzędzi dostarczanych w ramach System Zdalnego Dostępu. Większość funkcjonalności dostarczana jest przez Portal PL-LAB, dlatego wszystkie eksperymenty przygotowywane i konfigurowane są z jego użyciem. Serwis przechowuje i udostępnia niezbędne informacje o konfiguracji i dostępności urządzeń eksperymentalnych, o poziomie wykorzystania pasma na interfejsach wyjściowych z węzła oraz aktualnie realizowanych eksperymentach. Bazując na tej informacji użytkownicy mają możliwość tworzenia i zgłaszania specyfikacji swoich laboratoriów, z uwzględnieniem rzeczywistego stanu zasobów. Po zgłoszeniu specyfikacji użytkownik ma możliwość monitorowania stanu konfiguracji. W czasie trwania eksperymentu użytkownik może uzyskać niezbędne informacje jak użytkować sieć badawczą.

Cykl życia każdego prowadzonego w systemie PL-LAB eksperymentu podzielony został na pięć etapów:

1. **Definicja eksperymentu**, w ramach której użytkownik określa ramy czasowe eksperymentu, temat doświadczenia, identyfikuje wymagane w eksperymencie urządzenia oraz definiuje topologie połączeń sieciowych z uwzględnieniem wymaganych przepustowości. Zasoby dostarczane przez infrastrukturę PL-LAB są współdzielone przez wszystkich użytkowników systemu, w związku z powyższym bardzo istotne jest określenie rozsądnych ram czasowych eksperymentu. Przyjęto, że minimalny czas badań użytkownika to jeden dzień, a czas rozpoczęcia eksperymentu nie może być wcześniejszy niż dwa dni od jego zgłoszenia. Osoba zgłaszająca eksperyment określa również listę osób uprawnionych do użytkowania tworzonego Laboratorium Użytkownika. Serwis PL-LAB został skonstruowany tak, aby maksymalnie uprościć sposób zgłaszania eksperymentu oferując dedykowane narzędzie do budowy topologii. Przykład stworzonej za jego pomocą topologii eksperymentu składającej się z dwóch ruterów MX240 i dwóch serwerach został zaprezentowany na rys. 6.




**Rysunek 6.** Topologia przykładowego eksperymentu w PL-LAB

2. **Rezerwacja zasobów i oczekiwanie na konfigurację**. Po zgłoszeniu specyfikacji eksperymentu w serwisie PL-LAB, system przystępuje do weryfikacji dostępności zasobów. Jeśli weryfikacja przebiegnie nieprawidłowo użytkownik zostanie poproszony o zmianę specyfikacji. W przypadku poprawnej weryfikacji serwis rezerwuje elementy infrastruktury PL-LAB, które są niezbędne do budowy sieci eksperymentalnej użytkownika. Z uwagi na to, że System Zdalnego Dostępu do PL-LAB umożliwia zgłaszanie eksperymentów z wyprzedzeniem, urządzenia nie są konfigurowane od razu i oczekują na nadejście momentu rozpoczęcia eksperymentu. Dwa dni przed startem eksperymentu PL-LAB Portal wysyła do obsługi technicznej PL-LAB informację o konieczności konfiguracji eksperymentu.
3. **Konfiguracja eksperymentu**. Obsługa techniczna PL-LAB po otrzymaniu sygnału do zestawienia sieci badawczej przystępuje do konfiguracji urządzeń zgodnie z żądaną specyfikacją eksperymentu. Konfiguracja urządzeń odbywa się za pomocą autorskich skryptów zainstalowanych na przełącznikach dostępowych. Konfiguracja polega na ustawieniu sieci VLAN i portów dla żądanego eksperymentu. Dedykowana sieć zarządzania pozwala na zdalny dostęp do urządzeń w celach konfiguracyjnych, natomiast RADIUS i LDAP umożliwiają uwierzytelnianie i autoryzację osób uprawnionych do korzystania z danego Laboratorium Użytkownika. Poprawne wykonanie skryptów równoważne jest z zakończeniem konfiguracji urządzeń, o czym zostaje powiadomiony właściciel rezerwacji.
4. **Uruchomiony eksperyment**. Użytkownicy uprawnieni do pracy w eksperymencie uzyskują dostęp do wskazanych w specyfikacji eksperymentu urządzeń PL-LAB i mogą bez przeszkód prowadzić swoje badania. Preferowaną drogą dostępu do interfejsów dostępowych i konfiguracyjnych urządzeń jest sieć i koncentratory VPN. Infrastruktura systemu PL-LAB zapewnia izolację eksperymentu od niekorzystnego wpływu innych eksperymentów i ruchu w sieci PL-LAB. System zabezpiecza również Laboratorium Użytkownika przed niepowołanym dostępem.
5. **Usuwanie eksperymentu**. Po osiągnieciu zdefiniowanego w eksperymencie czasu jego zakończenia, serwis PL-LAB Portal przystępuje do zwolnienia zarezerwowanych zasobów. W sposób automatyczny przekonfigurowane są przełączniki dostępowe w celu usunięcia tych części konfiguracji, które zostały w wprowadzone na potrzeby danego eksperymentu. Wykorzystywane urządzenie zwracane są do puli urządzeń dostępnych i mogą zostać wykorzystane w następnych eksperymentach.

### 4.3 Wsparcie dla obsługi technicznej PL-LAB

Głównym założeniem tworzonej sieci PL-LAB była pełna niezależność poszczególnych laboratoriów wchodzących w jej skład. W dużej mierze wynika to z unikalnej specyfiki każdego z węzłów, jak również odmiennych wymagań użytkowników. Każdy z węzłów posiada różnorodną listę zainstalowanego sprzętu, jak również topologię połączeń sieciowych. W celu efektywnego zarządzania tak rozproszonym i heterogenicznym środowiskiem, powołano specjalną grupę obsługi

technicznej, w której skład weszły osoby z każdej jednostki utrzymującej węzeł sieci PL-LAB. Dla każdego z węzłów określono osoby pełniące funkcje:

- Opiekuna węzła - osoba o tej roli funkcyjnej odpowiedzialna jest za techniczną obsługę i utrzymanie prawidłowej pracy węzła PL-LAB. W przypadku problemów technicznych może zostać poproszona o podłączenie nowych urządzeń, zmianę połączeń czy fizyczny restart sprzętu. W portalu PL-LAB głównym zadaniem Operatora węzła jest utrzymania spójności informacji o fizycznej konfiguracji węzła, do czego posiada on niezbędne uprawnienie (dostępność perspektywy opiekun węzła).
- Opiekuna użytkowników - osoba o tej roli funkcyjnej odpowiedzialna jest za zarządzanie uprawnieniami do korzystania z systemu PL-LAB dla użytkowników wywodzących się z danej jednostki utrzymującej węzeł PL-LAB. Akceptując zgłoszenia rejestracyjne użytkownika osoba ta bierze odpowiedzialność za jego działania w systemie PL-LAB. W portalu PL-LAB Opiekun użytkowników posiada niezbędne uprawnienia (dostępność perspektywy opiekuna użytkowników) do akceptacji, blokowania lub nadawania wyższych uprawnień dla kont użytkowników danego węzła.

Do obsługi eksperymentów prowadzonych przez użytkowników PL-LAB powołano niezależny zespół opiekunów eksperymentów, którzy odpowiedzialni są za konfigurację oraz nadzór poprawnego funkcjonowania środowisk eksperymentalnych tworzonych w systemie PL-LAB.

Bez dodatkowych narzędzi praca powołanego zespołu byłaby mocno utrudniona a wraz z rozrastaniem się infrastruktury i większą liczbą eksperymentów wręcz niemożliwa. W czasie tworzenia specyfikacji Systemu Zdalnego Dostępu do sieci PL-LAB uwzględniono wymagania obsługi technicznej, kładąc duży nacisk na opracowanie mechanizmów zbierania informacji o stanie infrastruktury oraz automatycznej jej konfiguracji. Bazując na specyfikacji stworzono bazę danych sprzętu zainstalowanego w sieci PL-LAB uwzględniającą logiczny podział zasobów infrastruktury na oddzielne laboratoria i przeznaczenie urządzeń. Dla każdego urządzenia stworzono kalendarze ich użytkowania, w celu określenia dostępność dla eksperymentów użytkowników. Ujednolicono wzorzec specyfikacji zgłaszanych eksperymentów jak również stworzono bazę danych do ich przechowywania.

W celu zarządzania stworzonymi bazami danych zaimplementowano niezbędne interfejsy po stronie portalu PL-LAB. Serwis wyposażono również w skrypty automatycznej konfiguracji sieci wykorzystywane przez obsługę techniczną do zarządzania infrastrukturą i tworzenia wirtualnych Laboratoriów Użytkowników. Tworząc narzędzia położono nacisk na to, aby jak najwięcej akcji wykonywanych przez obsługę techniczną odbywało się z użyciem portalu PL-LAB. Głównymi przyczynami takiego podejścia była chęć jak największego uproszczenia procesu zarzadzania infrastrukturą oraz konieczność zachowania spójności miedzy fizyczną konfiguracją urządzeń a jej odzwierciedleniem w bazach danych systemu. Przy tworzeniu narzędzi uwzględniono logiczny podział infrastruktury na poszczególne jej węzły, ograniczając odpowiednio możliwości zarządzania dla po-

szczególnych uprawnień użytkowników. I tak dla Opiekunów węzła i Opiekunów użytkowników portal PL-LAB oferuje następujące funkcje zarządzania węzłem:

- Zarządzanie konfiguracją węzła - portal udostępnia szereg funkcji oraz podręczników użytkownika umożliwiających pełną konfigurację i reprezentację węzła w bazach danych System Zdalnego Dostępu. W tym pozwala określić podstawowe informacje dotyczące węzła (np. adres jego lokalizacji), jego dostępność dla eksperymentów użytkowników, konfigurację podsieci IPv4 i IPv6 dla interfejsów zarzadzania urządzeń w węźle oraz konfigurację koncentratora VPN.
- Zarządzenie urządzeniami zainstalowanymi w węźle - portal udostępnia szereg funkcji umożliwiających reprezentację w bazie danych Systemy Zdalnego Dostępu fizycznych urządzeń zainstalowanych w węźle, oraz udostępnianie ich na potrzeby eksperymentów użytkowników bądź na potrzeby zarządzania infrastrukturą PL-LAB. Dla zapewnienia jak największej elastyczności stworzone narzędzia pozwalają modelować dowolne typy urządzeń. Dla każdego z urządzeń istnieje możliwość specyfikacji jego parametrów technicznych, konfiguracji portów sieciowych, adresacji IP, sposobów dostępu i uwierzytelniania.
- Zarządzanie topologią połączeń węzła i interfejsami wyjścia z węzła - portal udostępnia szereg funkcji umożliwiających odzwierciedlanie w bazie systemu fizycznej topologii połączeń sieciowych węzła oraz jego interfejsów wyjściowych do sieci VPLS. Jako uproszczenie przyjęto, że każde urządzenie nie będące przełącznikiem dostępowym może być dołączone wyłącznie do przełącznika dostępowego. Umożliwiono również opiekunom węzła definiować na przełącznikach dostępowych porty wyjściowe z węzłów wraz z przypisanymi im sieciami VPLS (spośród wcześniej zdefiniowanych przez administratorów).
- Zarządzane kontami użytkowników - portal udostępnia szereg funkcji umożliwiających zarządzania kontami użytkowników danego węzła, w tym poziomem przyznanych uprawnień, blokowaniem/odblokowywaniem oraz usuwaniem kont użytkowników.

Oprócz zasobów dostarczanych przez poszczególne węzły laboratoryjne, System Zdalnego Dostępu udostępnia również zasoby wspólne dla całej infrastruktury. Są to między innymi sieci VPLS skonfigurowane w ramach sieci naukowej PIONIER, używane do zapewniania komunikacji międzywęzłowej, oraz VLAN-y używane to do konfiguracji wirtualnych sieci laboratoryjnych. Dla zarządzania i modelowania tych zasobów udostępniono administratorom dodatkowe funkcje w portalu PL-LAB. Mają oni możliwość definiowania dowolnych sieci VPLS, które następnie mogą być wykorzystane przy tworzeniu specyfikacji eksperymentów międzywęzłowych. Fizyczna konfiguracja sieci VPLS odbywa się poza siecią PL-LAB w ramach sieci PIONIER. Administratorzy mają również możliwość definicji VLAN-ów użytkowanych w systemie, zarówno tych zestawianych na potrzeby sieci PL-LAB (np. główny VLAN zarządzania PL-LAB), jak również używanych na potrzeby tworzenia eksperymentów.

Portal PL-LAB oferuje także wsparcie dla opiekunów eksperymentów poprzez nadzór nad zgłoszonymi i skonfigurowanymi eksperymentami pod kątem podjęcia odpowiednich akcji konfiguracyjnych (konieczność stworzenia czy usunięcia wirtualnej sieci eksperymentalnej). Z drugiej strony dostarcza skrypty umożliwiające automatyczną konfigurację środowisk badawczych użytkowników.

### 4.4 Rozwój i nowe funkcjonalności

W celu spełnienia nowych wymagań, które pojawiły się w trakcie użytkowania sieci PL-LAB, prowadzone są prace mające na celu rozbudowanie infrastruktury o dodatkowe funkcjonalności, takie jak:

- dostęp do Laboratorium Użytkownika poprzez bramę publiczną;
- wsparcie dla wirtualizacji serwerów.

Ponieważ koncentratory VPN umożliwiają dostęp tylko i wyłącznie do sieci zarządzania, w pierwszej wersji systemu nie było możliwości podłączenia się z sieci publicznej do sieci stworzonej w ramach eksperymentu. W toku prowadzonych badań użytkownicy zgłaszali jednak potrzebę dostępu do wirtualnych sieci eksperymentalnych. Aby to umożliwić, stworzona została dodatkowa usługa udostępniania bramy publicznej. Na żądanie użytkownika przygotowywany jest dostęp do maszyny wirtualnej posiadającej z jednej strony interfejs sieciowy w publicznej sieci, z drugiej zaś w jednym z VLAN-ów zdefiniowanych w Laboratorium Użytkownika. Ze względów bezpieczeństwa nie jest możliwe wpięcie bramy publicznej do sieci zarządzania PL-LAB.

Drugim istotnym usprawnieniem, nad wprowadzeniem którego trwają prace, jest automatyzacja tworzenia i zarządzania maszynami wirtualnymi. Obecnie użytkownicy mogą rezerwować na potrzeby eksperymentów wyłącznie maszyny (serwery) fizyczne, co nie zawsze jest efektywne, ponieważ zdarza się, że dany eksperyment nie wymaga dużej mocy obliczeniowej, podczas gdy na jego użytek rezerwowany jest cały serwer. Dlatego też trwają prace nad rozbudowaniem środowiska PL-LAB o umożliwienie rezerwacji zasobów obliczeniowych na poziomie maszyn wirtualnych, co wymaga rozbudowania portalu o funkcje zarządzania zasobami serwerów (liczbą zarezerwowanych procesorów, pamięci, miejsca na dysku, wirtualnymi interfejsami sieciowymi). Ponadto funkcjonalność ta wymagać będzie przeprowadzenia dodatkowych czynności konfiguracyjnych podczas tworzenia Laboratorium Użytkownika. Aby ułatwić zarządzanie zasobami serwerowymi rozważane jest wdrożenie oprogramowania zarządzającego chmurami obliczeniowymi (np. OpenStack [10]).

## 5 Podsumowanie

Artykuł prezentuje portal PL-LAB, który ma na celu udostępnienie narzędzi do usprawnienia procesu zarządzania infrastrukturą oraz procesu aplikowania o zasoby sieciowe w rozproszonej sieci badawczej PL-LAB. Główna funkcjonalność portalu związana jest z definiowaniem i zgłaszaniem przez użytkowników, którzy

chcą przeprowadzić eksperymenty, projektów tzw. Laboratoriów Użytkownika, czyli logicznych sieci laboratoryjnych wydzielonych z dostępnej w PL-LAB infrastruktury fizycznej. Z kolei osoby pełniące rolę obsługi technicznej sieci PL-LAB mogą za pomocą portalu przeprowadzić szereg czynności konfiguracyjnych i administracyjnych, jak np. zestawienie zdefiniowanego przez użytkownika wirtualnego laboratorium czy zarządzanie kontami użytkowników zarejestrowanych w systemie.

Obecnie dostęp do serwisu zarezerwowany jest wyłącznie dla naukowców przeprowadzających badania realizowane w ramach projektu IIP, jednakże po jego zakończeniu (lipiec 2013 r.) portal PL-LAB zostanie również udostępniony społeczności naukowo-badawczej niezaangażowanej w prace projektu IIP.

## Literatura

1. R. Krzywania et al. „PL-LAB: Polish Future Internet distributed laboratory", in the Proc. of 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing SNPD 2012, August 8-10, 2012, Kyoto, Japan
2. J. Śliwiński, R. Krzywania, Ł. Dolata, „PL-LAB: sieć eksperymentalna projektu Inżynieria Internetu Przyszłości", Przegląd Telekomunikacyjny, nr. 8-9, 2011, ISSN 1230-3496 (KSTiT'2011)
3. Projekt POIG.01.01.02-00-045/09-00 Inżynieria Internetu Przyszłości, `https://www.iip.net.pl`
4. PIONIER Polish Optical Internet. Strona www: `http://www.pionier.net.pl/online/pl/`
5. IEEE Std. 802.1Q-2011, IEEE Standard for Local and metropolitan area networks– Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks
6. Ł. Dolata i in. „System Zdalnego Dostępu do Sieci Eksperymentalnej PL-LAB", Przegląd Telekomunikacyjny, Wiadomości Telekomunikacyjne, 2012, R. 85, nr 8/9, pp. 1409-1417
7. P. Sezgedi et al., "With Evolution for Revolution: Managing FEDERICA for Future Internet Research", IEEE Communications Magazine Vol.47 No.7 pp. 34-39
8. The OneLab Initiative. Strona www projektu: `http://www.onelab.eu/`
9. PlanetLab. Strona www projektu: `http://www.planet-lab.org`
10. OpenStack Cloud Software: `http://www.openstack.org/`

# Scenariusze biznesowe w środowisku IPv6

Jordi Mongay Batalla, Piotr Krawiec

Instytut Łączności – Państwowy Instytut Badawczy

**Streszczenie** Artykuł ma na celu prezentację scenariuszy biznesowych, których możliwość realizacji, wydajność, łatwość implementacji bądź użytkowania wzrasta w przypadku, kiedy są one wdrażane w środowisku sieciowym korzystającym z nowszej wersji protokołu IP, tj. wersji 6, zamiast szeroko obecnie wykorzystywanego protokołu IP w wersji 4. Przedstawione scenariusze obejmują usługę telewizji internetowej, bezpośredniego łącza do pobieranego pliku, wykorzystania protokołu Point-to-Point, przetwarzania w chmurze, jak również usługi oferowane w ramach Internetu Rzeczy. Każdy z tych scenariuszy czerpie korzyści z funkcjonalności oferowanych przez IPv6, a niedostępnych w IPv4, takich jak olbrzymia przestrzeń adresowa, mechanizm autokonfiguracji, wsparcie dla mobilności oraz możliwość realizacji połączeń typu mulitcast.

## 1 Wprowadzenie

Pełniąca nadrzędną rolę w procesie przydzielania adresów IP organizacja IANA (ang. *Internet Assigned Numbers Authority* [1]) przekazała w lutym 2011 roku ostatnie wolne bloki adresów IPv4 do Regionalnych Rejestrów Internetowych RIR (ang. *Regional Internet Registry*). Z kolei we wrześniu 2012 pełniąca rolę europejskiego RIR organizacja RIPE NCC (ang. *RIPE Network Coordination Centre*) poinformowała, że rozpoczęła proces przydzielania adresów IPv4 z ostatniego posiadanego wolnego bloku /8 [2]. Możemy zatem stwierdzić, iż kwestia wyczerpania się puli dostępnych adresów IPv4, jeszcze kilka lat temu często traktowana jako odległe, wręcz mało realne zagrożenie, w chwili obecnej staje się faktem.

Jednym z działań mających na celu zapobieżenie temu problemowi było opracowanie nowej wersji protokołu IP, nazwanej IPv6, której pierwsze dokumenty specyfikacyjne zostały opublikowane już w roku 1998 (IETF RFC 2460 [3]). W chwili obecnej liczba dokumentów RFC związanych z IPv6 przekracza 200 ([4]), obejmując różnorodne aspekty, takie jak mobilność (RFC 6275), ruting (np. RFC 5308, 5340) czy bezpieczeństwo (np. RFC 6092). Pomimo tego w Polsce protokół IPv6 wciąż nie został powszechnie wdrożony do sieci. Zazwyczaj jest on już obsługiwany przez węzły sieci szkieletowych, jednakże kluczowym jest brak wsparcia dla IPv6 w sieciach dostępowych. Część dużych dostawców Internetu w Polsce oferuje co prawda natywny dostęp do IPv6, ale wyłącznie dla klientów korporacyjnych [5].

Powody takiej sytuacji są głównie natury ekonomicznej. Zmiana wykorzystywanej w sieci wersji protokołu IP z IPv4 do IPv6 generuje zazwyczaj ogromne

koszty związane z koniecznością wymiany na nowe wszystkich urządzeń niewspierających IPv6, przeszkolenia pracowników oraz dostosowania używanego oprogramowania. Ten ostatni czynnik wydaje się być szczególnie istotnym, ponieważ wiele narzędzi stosowanych w systemach operatorskich, jak np. systemy zarządzania relacjami z klientami CRM (ang. *Customer Relationship Management*) czy systemy wspomagające zarządzanie infrastrukturą teleinformatyczną, zostały stworzone od podstaw dla konkretnego operatora w postaci dedykowanych aplikacji zawierających wiele niestandardowych rozwiązań, stąd też rozszerzenie ich funkcjonalności o obsługę protokołu IPv6 może być procesem złożonym, a przez to kosztownym.

Biorąc pod uwagę tempo wdrażania protokołu IPv6 w sieciach operatorów wydaje się, iż w ich opinii proces migracji do IPv6 nie jest w pełni ekonomicznie uzasadniony, i potencjalne zyski wynikające z uruchomienia nowej wersji protokołu IP nie pokryją poniesionych nakładów. Operatorzy nie znajdują tzw. przełomowych aplikacji (ang. *killer applications*), działających jedynie w środowisku IPv6, które mogłoby skłonić użytkowników do korzystania z usług operatora oferującego natywny dostęp IPv6. W związku z tym operatorzy starają się jak najdłużej wykorzystać posiadaną infrastrukturę IPv4 przyjmując, że ogromna większość użytkowników nie zwraca uwagi na to, którą wersję protokołu IP używa korzystając z zasobów Internetu w sytuacji, gdy niezależnie od wersji protokołu dostępne są dla użytkownikom identyczne usługi. Z kolei fakt łączenia się użytkowników z Internetem głównie z wykorzystaniem protokołu IPv4 powoduje, że również dla dostawców usług i treści proces przystosowania do współpracy z IPv6 posiadanej przez nich infrastruktury i oprogramowania nie charakteryzuje się najwyższym priorytetem.

Zaprezentowany artykuł ma na celu przedstawienie wybranych scenariuszy biznesowych wykorzystujących własności protokołu IPv6, które to scenariusze są trudne do realizacji, o ile w ogóle możliwe, w środowisku korzystającym wyłącznie z protokołu IPv4. Tym samym mogą być one traktowane jako wartość dodana, premiująca operatorów sieci z wdrożonym protokołem IPv6 względem podmiotów posiadających wyłącznie sieć IPv4.

Plan artykułu jest następujący. W rozdziale drugim zaprezentowane zostały cechy protokołu IPv6 mogące być podstawą do zaoferowania nowych scenariuszy biznesowych. Następne rozdziały przedstawiają, korzystające z ww. własności, przykładowe aplikacje i scenariusze użycia. Są to: telewizja internetowa IPTV (rozdział 3), bezpośrednie łącze do pobieranego pliku (rozdział 4), protokół Point-to-Point (rozdział 5), przetwarzanie w chmurze (rozdział 6) oraz Internet Rzeczy (rozdział 7). Artykuł zakończony jest krótkim podsumowaniem.

## 2 Protokół IPv6 - podstawowe cechy

Protokół IPv6 jest rozwiązaniem niekompatybilnym ze swoim poprzednikiem, przez wzgląd na m.in. inny format nagłówka bądź też odmienne mechanizmy pozyskiwania adresów warstwy łącza danych innych węzłów w sieci. Jedną z głównych cech wyróżniających protokół IPv6 od IPv4 jest czterokrotnie dłuższe pole

adresowe, pozwalające (teoretycznie) zaadresować ok. $3{,}4 \cdot 10^{38}$ urządzeń, a zatem ponad $10^{28}$ razy więcej niż to jest możliwe w przypadku poprzedniej wersji protokołu IP. Wprowadza to dużą elastyczność w procesie przydziału pul adresowych: minimalny przydzielany obecnie prefiks to /64, co umożliwia wyróżnienie $2^{64}$ sieci ($\approx 1{,}7 \cdot 10^{19}$), z których w każdej można zaadresować do $2^{64}$ hostów (a zatem $4{,}3 \cdot 10^{9}$ więcej niż wynosi cała przestrzeń adresowa IPv4). Ułatwia to proces zarządzania sieciami, jak również zwiększa efektywność przekazu pakietów przez węzły sieciowe dzięki możliwości hierarchicznej agregacji ścieżek w sieci. Możliwość przydzielenia unikalnego adresu każdemu z urządzeń dołączonych do sieci niweluje konieczność stosowania mechanizmów translacji adresów sieciowych NAT (ang. *Network Address Translation*), co ma szereg zalet. Poprzez wyeliminowanie urządzeń NAT z sieci zwiększa się szybkość realizacji połączeń oraz zmniejsza zużycie energii, jak również przywraca pierwotną koncepcję bezpośredniej komunikacji koniec-koniec pomiędzy dwoma użytkownikami podłączonymi do sieci w dowolnym jej miejscu.

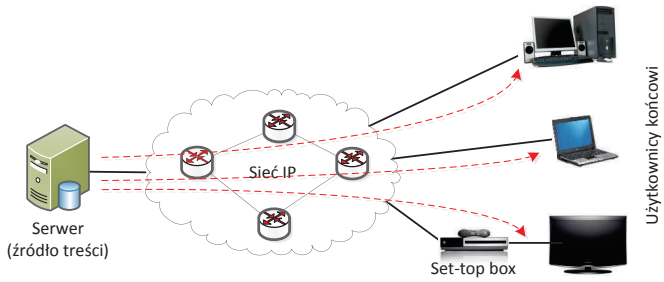
Kolejna cecha to mechanizmy autokonfiguracji [6] umożliwiające terminalowi w sposób w pełni automatyczny, bez ingerencji użytkownika bądź stosowania dedykowanych serwerów DHCP, dołączyć się do sieci i uzyskać wszystkie informacje niezbędne do komunikacji z innymi urządzeniami (tj. prefiks podsieci, własny adres IP, adres IP bramy sieciowej oraz adres serwera DNS [7]). Funkcjonalność ta w znaczący sposób upraszcza proces konfiguracji i zarządzania sieciami, co ma istotne znaczenie w przypadku sieci dostępowych obejmujących dużą liczbę podłączonych urządzeń końcowych.

Nieodłączną cechą protokołu IPv6 jest również wsparcie dla komunikacji typu multicast. W porównaniu do IPv4, IPv6 wprowadza większą liczbę dostępnych adresów mulitcast, możliwość łatwego zarządzanie zasięgiem połączenia (poprzez pole scope zawarte w adresie), jak również efektywniejszą realizację połączeń typu multicast w środowisku wielodomenowym [8]. Z kolei wbudowane w IPv6 mechanizmy wsparcia mobilności niwelują problemy, jakie występowały w przypadku obsługi mobilności użytkowników korzystających z protokołu IPv4, tj. zwiększenie opóźnienia pakietów w związku z ich przekazem poprzez węzeł sieci macierzystej, tzw. agenta domowego (ang. *Home Agent*) oraz wymiana dużej liczby wiadomości sygnalizacyjnych wynikających ze zmiany lokalizacji terminala [9].

Przedstawione powyżej własności protokołu IPv6 pozwalają na wdrożenie zaprezentowanych w następnych rozdziałach scenariuszy biznesowych, których realizacja w sieci IPv4 jest zazwyczaj skomplikowana i mało efektywna, a w niektórych przypadkach niemożliwa do przeprowadzenia.

## 3 Scenariusz 1: telewizja internetowa

Telewizja internetowa IPTV (ang. *Internet Protocol Television*) staje się jedną z najpopularniejszych usług wykorzystywanych w Internecie. Wyniki badań przedstawione w raportach firmy Ericsson ilustrują, iż połowa respondentów korzysta z IPTV przynajmniej raz w tygodniu [10], 35% zaś stwierdziło, że nie widzi




**Rysunek 1.** System IPTV

obecnie potrzeby zakupu bądź wypożyczania płyt DVD/Blue-ray, gdyż wszystkie treści multimedialne, z którymi chcą się zapoznać, są udostępnione w Internecie [11].

Rys. 1 prezentuje ogólną architekturę systemu IPTV. Serwer będący źródłem treści generuje strumień pakietów IP przenoszących sygnał wideo, który jest odbierany przez użytkowników końcowych. W sytuacji odbioru treści multimedialnych na urządzeniach które nie współpracują z protokołem IP (np. telewizor), konieczne jest zastosowanie urządzeń set-top box, dekodujących strumień pakietów IP na sygnał telewizyjny.

Telewizję internetową możemy podzielić na dwie grupy usług: 1) wideo na żądanie VoD (ang. *Video on Demand*) oraz 2) tzw. "telewizję na żywo"(ang. *live TV*). VoD pozwala użytkownikom dokonywać wyboru co i kiedy chcą oglądać, realizowana zaś jest w postaci zestawianych na żądanie połączeń punkt-punkt (unicast) pomiędzy użytkownikami a źródłem treści. Z kolei *live TV* związana jest z rozpoczęciem o określonej porze transmisji typu multicast skierowanej do grupy użytkowników zainteresowanych odbiorem danej treści wideo.

O ile wdrożenie w sieci IPv4 usługi VoD bazującej na połączeniach typu unicast nie nastręcza trudności, realizacja w takiej sieci usługi *live TV* jest problematyczna. Usługa ta wymaga przekazu danych z zapewnieniem reżimu czasu rzeczywistego, co uzyskuje się poprzez zastosowanie protokołów RTP/RTCP (ang. *Real-time Transport Protocol / Real-time Transport Control Protocol* [12]). Niestety, ograniczona liczba dostępnych publicznych adresów IPv4 wymusza stosowanie w obecnym Internecie mechanizmów NAT, przez co nie jest zapewniona bezpośrednia łączność pomiędzy użytkownikiem końcowym a serwerem strumieniującym. Wymaganie to jest szczególnie ważne w sytuacji, gdy protokół RTP jest obsługiwany w warstwie transportowej przez bezpołączeniowy protokół UDP. Mimo iż opracowane zostały rozwiązania pozwalające na przejście strumienia poprzez NAT z zachowaniem łączności koniec-koniec (np. *Session Traversal Utilities for NAT* - STUN [13] czy *Traversal Using Relays around NAT* - TURN [14]), to mają one zastosowanie dla konkretnych, ściśle określonych przy-

padków, zależnych m.in. od użytej implementacji mechanizmu NAT (którego działanie nie jest w pełni zestandaryzowane), brak jest natomiast jednej uniwersalnej metody działającej w każdej sytuacji. Dlatego też obecnie wdrożone systemy IPTV wykorzystujące protokół RTP obejmują zazwyczaj jedną domenę i przeznaczone są jedynie dla obsługi użytkowników danego operatora, zaś w wielodomenowym środowisku Internetu do przekazu treści multimedialnych stosuje się głównie połączeniowy protokół TCP.

Zastosowanie protokołu IPv4 implikuje także problem związany z realizacją połączeń multicast. Transmisja multicast w przypadku usługi *live TV*, zakładającej odbiór w danym momencie tej samej treści przez wielu użytkowników, umożliwia optymalne wykorzystanie zasobów takich jak pojemności łączy, gdyż na współdzielonych fragmentach ścieżek jest ona realizowana w postaci pojedynczego strumienia pakietów, jak i obciążenia serwerów strumieniujących, które muszą obsłużyć tylko jedno połączenie niezależnie od liczby dołączonych użytkowników. Niestety w celu realizacji usługi multicast w środowisku IPv4 konieczne jest rozszerzenie funkcjonalności węzłów sieciowych o dodatkowe mechanizmy zlokalizowane powyżej warstwy sieci. Z tego względu multicast IPv4 realizowany jest zazwyczaj w tzw. scenariuszu *walled garden*, a więc na ograniczonym obszarze, zamkniętym"w obrębie domeny danego dostawcy usług. Ponadto wspomniane powyżej dodatkowe mechanizmy nie współpracują z modułami warstwy sieciowej, przez co nie jest możliwe skuteczne zastosowanie mechanizmów poprawiających efektywność transmisji na poziomie sieciowym, takich jak mechanizm adaptacji na poziomie sieci NAL (ang. *Network Level Adaptation*) [15].

Wydajna, działająca także w scenariuszu wielodomenowym platforma IPTV może być natomiast w łatwy sposób wdrożona w sieci IPv6. Likwidacja urządzeń NAT dzięki dużej liczbie dostępnych adresów IPv6 przywraca bezpośrednią komunikację koniec-koniec w obrębie całego Internetu, przez co możliwe jest zastosowanie dedykowanych dla transmisji strumieniowej w czasie rzeczywistym protokołów RTP/RTCP bazujących na bezpołączeniowym protokole UDP. Ponadto protokół IPv6 cechuje obligatoryjna implementacja mechanizmów multicast w węzłach sieci, przez co usługa *live TV* może być w łatwy sposób realizowana w postaci połączeń multicast zestawianych na poziomie warstwy sieciowej. Dzięki temu możliwe jest wykorzystanie mechanizmów adaptacyjnych na poziomie warstwy sieci, jak np. NAL, oraz algorytmów dynamicznego dopasowania topologii drzew multicast'owych uwzględniających warunki panujące w sieci, przez co wzrasta efektywność wykorzystania zasobów sieciowych. Należy podkreślić, że realizacja usługi IPTV w środowisku IPv6 pozwala na przeprowadzenie adaptacji do aktualnie panujących warunków w różnych punktach sieci, a nie wyłącznie na końcach połączenia serwer-użytkownik, jak to jest w przypadku sieci IPv4, gdzie adaptacja występuje wyłącznie w warstwie aplikacji. Własność ta ma szczególne znaczenie w sytuacji, gdy usługa IPTV realizowana jest w scenariuszu wielodomenowym.

Kolejną zaletą bezpośredniej komunikacji koniec-koniec, dostępnej dzięki wdrożeniu protokołu IPv6, jest zapewnienie możliwości komunikacji pomiędzy operatorem usługi IPTV a modułami set-top box użytkowników. W ten sposób

operator usługi może wdrożyć mechanizmy mające na celu zwiększenie jakości świadczenia usług dla użytkowników, takie jak: 1) monitorowanie jakości przekazu strumienia danych bezpośrednio w punkcie końcowym, 2) zdalna konfiguracja urządzenia dopasowana do potrzeb użytkownika, 3) badanie preferencji użytkownika odnośnie wyboru treści multimedialnych, 4) dostosowanie informacji reklamowych do jego zainteresowań, itp.

## 4 Scenariusz 2: bezpośrednie łącze do pobieranego pliku

Następnym scenariuszem, który może wykorzystać własności protokołu IPv6, jest usługa tzw. bezpośredniego łącza do pobieranego pliku DDL (ang. *Direct Download Link*). Usługa DDL jest stosowana w celu udostępniania plików poprzez publikację łącza internetowego wskazującego na jedną (w przeciwieństwie do systemów peer-to-peer) lokalizację w sieci, z której użytkownik może ściągnąć dany plik.

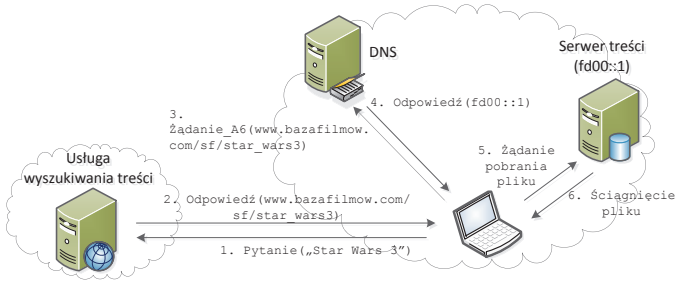
Realizacja usługi DDL w sieci IPv6 daje możliwość zastosowania dodatkowych mechanizmów opracowanych dla systemu DNS (ang. *Domain Name System*) w celu obsługi adresów IPv6 [16]. Poprzez wykorzystanie typu rekordu A6 możemy dopisać unikalny lokalny adres IPv6 do identyfikatora URL (ang. *Uniform Resource Locator*) wskazującego dany plik. Dzięki temu kopia tego pliku może zostać umieszczona na serwerach rozproszonych po całej sieci, zaś serwery DNS będą tłumaczyły odnoszące się do pliku identyfikatory URL nie tylko na globalny adres IPv6 serwera, na którym pierwotnie został on zapisany, ale również na adresy IPv6 lokalnych serwerów przechowujących jego kopie.

Przykład ten ilustruje rys. 2. Przyjmijmy, że użytkownik chce pobrać z sieci plik z filmem Śtar Wars 3". Korzystając z usługi wyszukiwania (krok 1) znajduje identyfikator URL poszukiwanego pliku (krok 2), który to identyfikator przekazuje następnie do swojego serwera DNS celem jego przetłumaczenia na adres IPv6 serwera, na którym plik jest umieszczony (krok 3). W przypadku gdy kopia poszukiwanego pliku została wcześniej umieszczona na lokalnym serwerze treści i odpowiedni wpis został dokonany w serwerze DNS obsługującym daną domenę, wówczas serwer DNS przetłumaczy URL na adres IPv6 lokalnego serwera (krok 4). W ten sposób użytkownik pobierze kopię pliku z lokalnego serwera treści (krok 5 i 6) zamiast z odległego serwera przechowującego oryginalny plik.

Powyższy mechanizm jest korzystny dla operatora, gdyż proces pobrania pliku realizowany jest w obrębie jego domeny, przez co nie generuje kosztów wynikających z transferu ruchu z domen innych operatorów (co nastąpiłoby w sytuacji, kiedy użytkownik ściągnąłby oryginalny plik przechowywany na serwerze zlokalizowanym w domenie innego operatora).

## 5 Scenariusz 3: protokół Point-to-Point

Protokół Point-to-Point (PPP) [17] jest protokołem warstwy łącza danych służącym do zestawienia połączenia typu punkt-punkt na poziomie warstwy drugiej modelu OSI. Mimo iż jego zastosowanie może być wielorakie, obecnie najczęściej




**Rysunek 2.** Realizacja usługi DDL w sieci IPv6

wykorzystywany jest przy podłączaniu terminali do Internetu za pomocą modemów oraz przy użyciu techniki DSL (ang. *Digital Subscriber Line*). Wynika to z faktu, iż PPP wymaga bezpośredniej komunikacji pomiędzy punktami końcowymi, co często nie jest możliwe do osiągnięcia w wielodomenowym środowisku IPv4, kiedy połączenia między urządzeniami realizowane są za pośrednictwem mechanizmów warstw wyższych, takich jak NAT czy firewall. Skutkuje to koniecznością jednoczesnego stosowania dodatkowych protokołów, jak na przykład protokołów kontrolnych warstwy sieciowej (ang. *Network Control Protocols*), co czyni użycie protokołu PPP w sieci Internet IPv4 bardziej skomplikowanym. W efekcie zastosowania tego protokołu ograniczone zostały jedynie do rozwiązań opartych na tunelowaniu lub typu *overlay* [18].

Wprowadzenie do Internetu protokołu IPv6, przywracającego bezpośrednią komunikację koniec-koniec, w znaczący sposób ułatwia tworzenie połączeń za pomocą protokołu PPP. W środowisku IPv6 protokół PPP czyni możliwym zestawianie łączy warstwy drugiej pomiędzy użytkownikami umiejscowionymi w dowolnych punktach sieci, które mogą być wykorzystane dla realizacji różnorodnych usług (np. połączeń VoIP).

## 6 Scenariusz 4: chmury obliczeniowe

Jednym z najbardziej popularnych obecnie trendów w obszarze technologii informacyjnych jest przetwarzanie w chmurze (ang. *cloud computing*). Wprowadziło ono nową koncepcję dostarczania oraz używania aplikacji przez użytkowników za pomocą usługi dostępu do zasobów obliczeniowych niezależnie od ich fizycznej lokalizacji. Przetwarzanie w chmurze daje użytkownikom możliwość wykupienia dostępu poprzez sieć do, zależnie od przyjętego modelu biznesowego, niezbędnych mu w danej chwili aplikacji bądź zasobów obliczeniowych. Podejście takie bardzo często jest rozwiązaniem efektywniejszym ekonomicznie niż bezpośredni zakup i następnie utrzymanie własnej infrastruktury teleinformatycznej. Jedną z największych zalet przetwarzania w chmurze jest duża elastyczność w dostępie

do zasobów. Przykładowo, zależnie od aktualnego zapotrzebowania, użytkownik może wykupić za daną kwotę moc obliczeniową 100 serwerów przez jedną godzinę albo jednego serwera przez 100 godzin, podczas gdy nabycie tradycyjnej infrastruktury instalowanej w siedzibie użytkownika wymaga, dla zapewnienia prawidłowego działania systemu, zakupu urządzeń o mocy wystarczającej dla obsługi szczytowego obciążenia, która to moc przez większość czasu mogłaby pozostać niewykorzystywana.

Realizacja chmury obliczeniowej w oparciu o sieć wykorzystującą protokół IPv6 przynosi następujące korzyści [19]. Ponieważ pula adresów IPv6 przydzielana operatorowi (zwykle jest to prefiks /48) pozwala zaadresować za pomocą unikalnych adresów globalnych olbrzymią liczbę maszyn wirtualnych tworzonych w ramach chmury, korzystający z niej użytkownicy mają zapewniony dostęp do przydzielonych im zasobów niezależnie od miejsca ich dołączenia do sieci (w obecnym Internecie IPv4 interfejsy maszyn wirtualnych adresowane są zazwyczaj przy wykorzystaniu puli adresów prywatnych, do których dostęp z innych domen wymaga użycia NAT [20]). Z kolei dostępne w IPv6 mechanizmy autokonfiguracji znacząco upraszczają zarządzanie rozległymi domenami, obejmującymi zarówno infrastrukturę fizyczną, jak i wirtualną. Protokół IPv6 ułatwia również proces zarządzania zasobami w chmurze poprzez uproszczenie takich operacji jak: 1) dynamiczna zmiana podsieci (migracja danej sieci wirtualnej na inną infrastrukturę fizyczną może skutkować modyfikacją wyłącznie tej części adresu IPv6 która określa prefiks sieci, podczas gdy pozostała część adresu identyfikująca host pozostaje niezmieniona), oraz 2) równoważenie obciążenia (ang. *workload rebalancing*) poprzez relokację działających maszyn wirtualnych wykorzystując mechanizmy obsługi wsparcia dla mobilności oferowane przez IPv6 [19].

## 7 Scenariusz 5: Internet Rzeczy

Internet Rzeczy (ang. *Internet of Things* - IoT) definiowany jest jako platforma sieciowa zapewniająca automatyczną (tj. bez ingerencji człowieka) komunikację pomiędzy fizycznymi i wirtualnymi obiektami. Obiekty te są ąktywne- przekazują sobie wzajemnie informacje, potrafią zidentyfikować wydarzenia oraz zmiany zachodzące w otaczającym je środowisku i zareagować na nie w odpowiedni sposób.

W porównaniu do obecnego Internetu, IoT cechuje się dużo większą liczbą urządzeń uczestniczących w procesie komunikacji. Jak prognozuje firma Ericsson [21], już w roku 2020 liczba obiektów podłączonych do Internetu będzie 10-krotnie większa niż cała przestrzeń adresowa dostępna w protokole IPv4. Powyższy problem braku dostatecznej liczby adresów IPv4 w przypadku Internetu Rzeczy nie może być zniwelowany za pomocą mechanizmów NAT, gdyż model komunikacyjny przyjęty dla IoT zakłada, że podłączone do Internetu urządzenia będą pełniły funkcje zarówno klienta wysyłającego zapytania do serwerów, jak i serwera obsługującego zgłoszenia otrzymywane od klientów (przykład: kontroler oświetlenia, wysyłający komendy sterujące do poszczególnych lamp, a zarazem obsługujący żądania wysyłane przez aplikacje sterujące oświetleniem). Należy również wziąć pod uwagę konieczność utrzymania połączenia koniec-koniec

poprzez NAT także w okresach, kiedy brak jest wymiany danych pomiędzy stronami. Generowane są wówczas dodatkowe wiadomości typu keepalive, co jest niekorzystne w przypadku komunikacji z urządzeniami o ograniczonych zasobach energetycznych (np. sensory z zasilaniem bateryjnym).

Dla scenariuszy z obszaru Internetu Rzeczy protokół IPv6 wydaje się być jedynym dostępnym obecnie rozwiązaniem pozwalającym na ich realizację przez wzgląd na dostępną przestrzeń adresową, jak również mechanizmy autokonfiguracji. O jego znaczeniu dla IoT może świadczyć fakt, że organizacje standaryzacyjne przyjmują IPv6 za podstawowy protokół sieciowy dla komunikacji pomiędzy urządzeniami M2M (ang. *machine-to-machine*), czyli głównego scenariusza komunikacyjnego występującego w IoT [22]. W ramach IETF opracowano protokół 6LoWPAN [23,**?**], który pozwala na przekaz pakietów IPv6 w bezprzewodowych sieciach sensorowych WSN (ang. *Wireless Sensor Networks*) standardu IEEE 802.15.4, dzięki czemu sensory mogą w sposób bezpośredni komunikować się z pozostałymi urządzeniami podłączonymi do Internetu. 6LoWPAN dodaje warstwę adaptacyjną powyżej warstwy łącza danych, która umożliwia efektywną transmisję pakietów IPv6 w oparciu o łącza radiowe standardu IEEE 802.15.4 (m.in. poprzez kompresję nagłówków warstwy transportowej i sieciowej w celu zmniejszenia narzutu transmisyjnego). Oprócz standardowego rutingu na poziomie warstwy IEEE 802.15.4 (a więc poniżej warstwy sieci), 6LoWPAN umożliwia również ruting IPv6 pomiędzy węzłami sieci sensorowej. Specjalny protokół rutingu IPv6 dla sieci WSN definiuje dokument RFC 6550 [25]. Protokół ten w swoim działaniu bierze pod uwagę ograniczone możliwości węzłów sieci WSN, tj. małą moc procesora i pojemności pamięci oraz bateryjne zasilanie. IETF prowadzi także prace standaryzacyjne nad protokołem CoAP (Constrained Application Protocol) [26], który ma na celu implementację w środowisku sieci WSN opartych na protokole 6LoWPAN rozproszonych aplikacji bazujących na technologii usług sieciowych (ang. *web services*). Dzięki opracowanym mechanizmom translacji, aplikacje wykorzystujące CoAP w łatwy sposób będą mogły komunikować się ze standardowymi aplikacjami sieciowymi opartymi na protokole HTTP.

Dotychczas zaproponowano wiele scenariuszy związanych z Internetem Rzeczy i protokołem IPv6, odnoszących się do różnorodnych dziedzin (rozrywki, edukacji, bezpieczeństwa itp.). Poniżej zaprezentowano dwa przykładowe scenariusze w obszarze opieki zdrowotnej oraz transportu:

- Aplikacja przeznaczona do stałego monitoringu stanu zdrowia pacjentów niezależnie od ich lokalizacji; dzięki zastosowaniu unikalnych globalnych adresów IPv6 operator usługi w łatwy sposób może skomunikować się z czujnikami monitorującymi parametry fizjologiczne pacjenta w celu ich konfiguracji bądź przeprowadzenia czynności konserwacyjnych (aktualizacja oprogramowania itp.), co jest szczególnie ważne w sytuacji kiedy monitorowaniu podlegają osoby starsze bądź niepełnosprawne.
- Aplikacja wykorzystująca mechanizmy wsparcia mobilności i transmisji typu multicast dostępne w IPv6 w celu zapewnienia komunikacji pomiędzy urządzeniami zainstalowanymi wzdłuż drogi a poruszającymi się po tej dro-

dze pojazdami (tzw. komunikacja typu Vehicle-to-Infrastructure V2I), jak również pomiędzy samymi pojazdami (Vehicle-to-Vehicle V2V); aplikacja ta może być użyta do automatycznego rozpowszechniania wzdłuż drogi komunikatów dotyczących bezpieczeństwa (np. informacja o innym pojeździe zbliżającym się do skrzyżowania) bądź usprawnieniu ruchu (przekazywanie wiadomości o występujących korkach lub innych utrudnieniach na drodze).

## 8 Podsumowanie

Wdrożenie protokołu IPv6 w Internecie wciąż pozostaje na niskim poziomie. Celem niniejszego artykułu było przedstawienie wybranych scenariuszy biznesowych, których realizacja w sieci IPv6 w znaczący sposób upraszcza procesy zarządzania, poprawia efektywność oraz wydajność i pozwala wdrożyć usługę w środowisku wielodomenowym, a nie wyłącznie w pojedynczej domenie kontrolowanej przez danego operatora, jak to zwykle ma miejsce w środowisku IPv4. Wynika to z takich własności protokołu IPv6 jak: ogromna przestrzeń adresowa, autokonfiguracja, obsługa mobilności użytkowników oraz połączeń typu mulitcast. Zaprezentowane scenariusze sugerują, iż również operatorzy których w chwili obecnej problem wyczerpania publicznych adresów IPv4 bezpośrednio nie dotyczy, mogą znaleźć ekonomiczne uzasadnienie dla wdrożenia protokołu IPv6 w zarządzanej przez nich sieci.

## Literatura

1. First The Internet Assigned Numbers Authority. Strona www: `http://www.iana.org/`
2. RIPE NCC IPv4 Exhaustion (dostęp: marzec 2013): `http://www.ripe.net/internet-coordination/ipv4-exhaustion`
3. S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", IETF RFC 2460, December 1998
4. "RFCs related to IPv6", informacja dostępna na stronie www (dostęp: marzec 2013): `http://ipv6now.com.au/RFC.php`
5. Kamil Folga, "IPv6 a sprawa polska". Networld nr 4/2012, 2012
6. S. Thomson, T. Narten, T. Jinmei, "IPv6 Stateless Address Autoconfiguration", IETF RFC 4862, September 2007
7. J. Jeong et al.,"IPv6 Router Advertisement Options for DNS Configuration", IETF RFC 6106, November 2010
8. Chris Metz and Mallik Tatipamula, "A Look at Native IPv6 Multicast", IEEE Internet Computing, vol. 8, no. 4, pp. 48-53, July/August, 2004
9. R. Gunasundari, S. Shanmugavel, "Performance comparison of mobile IPv4 and mobile IPv6 protocols in wireless systems", Proceedings of the First international conference on COMmunication Systems And NETworks COMSNETS'09, pp. 83-90, IEEE Press Piscataway, NJ, USA, 2009, ISBN: 978-1-4244-2912-7
10. "TV & Media Consumer Insights", Raport techniczny, Ericsson ConsumerLab, 2011, dostępny on-line: `http://www.ericsson.com/res/docs/2011/tv_and_media.pdf` (dostęp - czerwiec 2012)

11. A. Erlandsson, N. Rönnblom, A. Ericson, "TV & Video 2011 - Consumer Trends", Raport techniczny, Ericsson ConsumerLab, 2011, dostępny on-line (dostęp - czerwiec 2012): http://www.ericsson.com/res/docs/2011/11_1650_rev_a.pdf
12. H. Schulzrinne et al., "RTP: A Transport Protocol for Real-Time Applications", IETF RFC 3550, July 2003
13. J. Rosenberg et al., "Session Traversal Utilities for NAT (STUN)", IETF RFC 5389, October 2008
14. R. Mahy et al., "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", IETF RFC 5766, April 2010
15. Stephan Wenger, "H.26L over IP: The IP-Network Adaptation Layer", in Proc. of Packet Video Workshop, 2002
16. M. Crawford, C. Huitema, "DNS Extensions to Support IPv6 Address Aggregation and Renumbering", IETF RFC 2874, July 2000
17. W. Simpson, "The Point-to-Point Protocol (PPP)". IETF RFC 1661, July 1994
18. L. Mamakos, "A Method for Transmitting PPP Over Ethernet (PPPoE)", IETF RFC 2516, February 1999
19. John Rhoton, "Optimizing Cloud Computing with IPv6", HP Technology Forum & Expo 2009, prezentacja dostępna online (dostęp - czerwiec 2012): http://ipv6.net/IPv6-presentations/optimizing-cloud-computing-with-ipv6-presentation.html
20. Wes Morgan, "IPv6's Killer App (Finally): Cloud Computing", IT WORLD, dostępny online (dostęp - czerwiec 2012): http://www.itworld.com/networking/85651/ipv6s-killer-app-finally-cloud-computing
21. Ericsson Press Release, "CEO to shareholders: 50 billion connections 2020", dostępny online (dostęp - czerwiec 2012): http://www.ericsson.com/news/1403231
22. "Machine-to-Machine Communication System Requirements. Stage 1 Requirements", Technical Specification Group S of the Third Generation Partnership Project 2 (3GPP2), May 2011
23. G. Montenegro et al., "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", IETF RFC 4944, September 2007
24. J. Hui, P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", IETF RFC 6282, September 2011
25. T. Winter, P. Thubert (eds.), "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", IETF RFC 6550, March 2012
26. Z. Shelby et al. "Constrained Application Protocol (CoAP) - draft-ietf-core-coap-11", IETF RFC Internet-Draft, July 16, 2012

# Zarządzanie mobilnością w nowoczesnych sieciach IP oraz sieciach Internetu Przyszłości

Józef Woźniak[1], Krzysztof Gierłowski[1], Tomasz Gierszewski[1], Michał Hoeft[1]

Katedra Teleinformatyki, Wydział Elektroniki, Telekomunikacji i Informatyki
Politechniki Gdańskiej

**Streszczenie** Streszczenie: W rozdziale przedstawiono krótką charakterystykę rynku teleinformatycznego i wymagań nowych usług sieciowych, porównując je z możliwościami funkcjonalnymi oferowanymi przez współczesne systemy i sieci, tak przewodowe jak i bezprzewodowe. Sformułowano ograniczenia architektury TCP/IP w aspekcie obsługi urządzeń mobilnych i nowych scenariuszy mobilności, biorąc pod uwagę zarówno oczekiwania użytkowników, jak i możliwości sprzętowe, oraz naturalne ograniczenia stosu protokołów IP. Na podstawie dokonanej oceny, przedstawiono rozwiązania protokolarne niezbędne dla realizacji usług pożądanych przez użytkowników mobilnych. Zaprezentowano też szereg modyfikacji i rozszerzeń klasycznych protokołów mobilności wykorzystujących mechanizmy implementowane w warstwie sieciowej. Ponadto, przedstawiono rozwiązania, w których mobilność wspierana jest przez wyższe warstwy stosu TCP/IP, a także wskazano na tendencje w rozwoju protokołów mobilności, dotyczące wyraźnego rozdziału komponentów adresowych identyfikujących użytkownika i jego bieżącą lokalizację. Analizy powyższej dokonano zarówno w kontekście współczesnej sieci IP jak i nowych propozycji Internetu Przyszłości.

## 1 Wprowadzenie

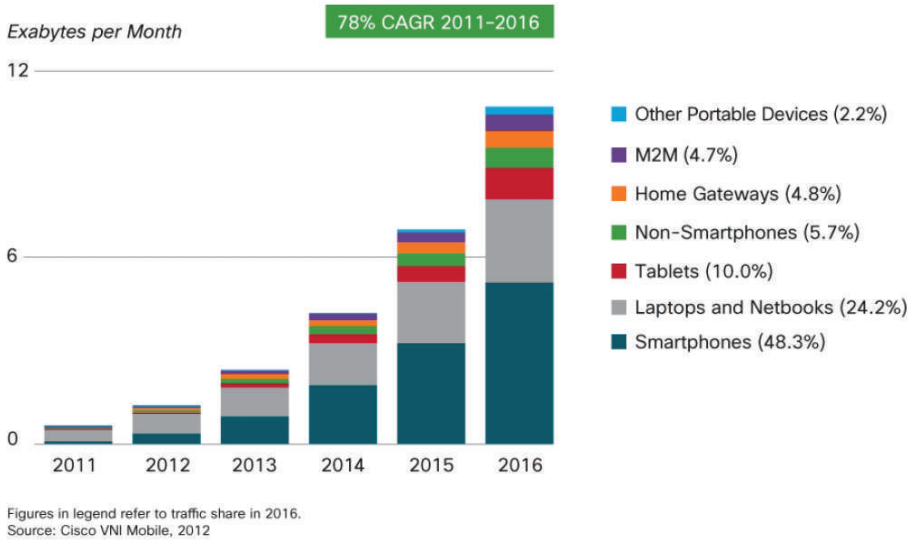
Szybki rozwój sieci IP i świadczonych przez nie usług spowodował pojawienie się nowych aplikacji i funkcjonalności dostępnych za pośrednictwem uniwersalnego protokołu IP i globalnego Internetu. Z kolei dynamiczny rozwój nowych generacji urządzeń przenośnych różnej klasy, zainspirował budowę nowych, głównie wąskopasmowych sieci bezprzewodowych. Rosnąca liczba wykorzystywanych laptopów, palmtopów, czy smartfonów - urządzeń mających możliwości obliczeniowe porównywalne z komputerami stacjonarnymi, sprawia jednak, że ich użytkownicy oczekują dostępności nowych usług szerokopasmowych również w środowisku mobilnym. Urządzenia te i ich użytkownicy, których liczba przekroczyła już liczbę użytkowników stacjonarnych, wymuszają zmiany we współczesnym Internecie, z których na szczególną uwagę zasługuje konieczność wprowadzenia wsparcia dla "mobilności". Mankamentem współczesnego Internetu jest jednak fakt, iż zastosowana w nim architektura TCP/IP została zaprojektowana z myślą wyłącznie o użytkownikach stacjonarnych. By rozwiązać ten problem i zapewnić efektywne zarządzanie mobilnością w powyższym środowisku, zaproponowano

szereg rozwiązań nakładkowych, wśród których najpopularniejsze są rozwiązania z rodziny Mobile IP. Nie są to niestety rozwiązania w pełni dostosowane do współczesnych potrzeb aplikacji i wzorców mobilności. Nieodzowne są zatem zarówno ich optymalizacje jak też opracowanie i wdrożenie nowych rozwiązań.

Dominująca rola Internetu, jako globalnej sieci teleinformatycznej, oferującej dostęp do różnorodnych zasobów i specjalizowanych usług, spowodowała zaadoptowanie protokołu IP i stosu TCP/IP, jako podstawowej i powszechnie akceptowanej metody organizacji pracy sieci. Wiąże się z tym pojęcie *All IP*, oznaczające implementację protokołu IP (*Internet Protocol*) i architektury TCP/IP (*Transmission Control Protocol/IP*) w praktycznie wszystkich systemach teleinformatycznych. Podejście powyższe umożliwia w efekcie stosunkowo łatwą i daleko posuniętą integrację systemów i sieci teleinformatycznych. Jednym z widocznych jej przejawów jest wyposażanie przenośnych urządzeń końcowych w wiele fizycznych interfejsów, co umożliwia im wykorzystanie różnorodnych technik dostępu do sieci, a zatem pracę w środowisku heterogenicznym. Efektem powyższego trendu jest też popularyzacja idei *Always Best Connected* (ABC) [1], związanej z możliwością wyboru przez użytkownika (bądź też automatycznie przez urządzenie końcowe) najlepszej z kilku dostępnych opcji połączenia sieciowego (najszybszej, najtańszej, etc.). Jednym ze sposobów realizacji koncepcji ABC może być wykorzystanie multihomingu, czyli jednoczesnego podłączenia terminala do kilku sieci. Pozwala to na realizację dodatkowej optymalizacji komunikacji terminala z otoczeniem [2].

Nie ma wątpliwości, że główną siłą napędową telekomunikacji w ostatnich dwóch dziesięcioleciach był segment technologii bezprzewodowych i nowe generacje urządzeń przenośnych. Na przestrzeni ostatnich lat obserwujemy bardzo szybki wzrost popularności różnorodnych mobilnych urządzeń przetwarzania i prezentacji danych cyfrowych. Do licznej grupy takich wielofunkcyjnych terminali można zaliczyć zarówno laptopy, palmtopy, PDA, smartfony, jak również wielofunkcyjne urządzenia nawigacji GPS (*Global Positioning System*), a nawet przenośne odtwarzacze mediów MP3/MP4. W obszarze zastosowań tych urządzeń można zaobserwować postępującą konwergencję - coraz więcej funkcji jest integrowanych w pojedynczym urządzeniu, dzięki czemu zwiększa się obszar ich zastosowań. Znakomita większość wielofunkcyjnych terminali umożliwia wykorzystanie protokołu IPv4, a coraz więcej także jego nowej wersji - IPv6. Powszechność stosowania protokołów IP stwarza możliwości tworzenia nowych, użytecznych usług, jak również nowego wykorzystania znanych rozwiązań. Szacuje się, że w ciągu kliku najbliższych lat ruch IP wygenerowany przez różne urządzenia mobilne znacząco wzrośnie (por. Rys. 1)

Cykliczne pojawianie się nowych generacji urządzeń mobilnych i ich rosnące możliwości techniczne pozwalają użytkownikom takich terminali korzystać z usług do tej pory zarezerwowanych wyłącznie dla tradycyjnych urządzeń stacjonarnych. Coraz większą popularność, w odniesieniu do terminali ruchomych, cieszą się np. usługi strumieniowania mediów. Należy uznać, iż obecny użytkownik urządzenia mobilnego oczekuje dostępu do równie rozbudowanego zbioru usług, co użytkownik stacjonarny. W związku z tym, pojawia się nowe wyzwanie zapew-




**Rysunek 1.** Szacowany wzrost liczby urządzeń sieciowych [3]

nienia pełnej dostępności urządzeń mobilnych oraz zagwarantowania ciągłości połączenia w trakcie ich przemieszczania.

Wspomniane powyżej tendencje, w tym rosnąca liczba terminali ruchomych oraz ogromny wzrost ruchu generowanego przez te urządzenia wymuszają konieczność implementacji protokołów pozwalających na dynamiczne zarządzanie stacjami mobilnymi podczas ich przemieszczania się między punktami przyłączenia do sieci. Możemy przy tym mówić o różnych scenariuszach mobilności - od mobilności terminala, poprzez mobilność użytkownika, aż do mobilności usług i sieci. Każdorazowo jednakże rozwiązanie problemu wsparcia mobilności związane będzie z implementacją właściwych mechanizmów i funkcjonalności w warstwowej architekturze TCP/IP. Proponując stosowne rozwiązania protokolarne musimy zmierzyć się z wyraźnym niedopasowaniem tej architektury do coraz powszechniejszych wymagań i oczekiwań użytkowników terminali mobilnych, pamiętając, że architektura Internetu i sieci IP została zaprojektowana z myślą o wyłącznie wymaganiach i możliwościach urządzeń stacjonarnych. W kolejnych podrozdziałach dokonamy krótkiego przeglądu popularnych klas usług dostępnych w sieciach IP oraz ich wymagań dotyczących jakości komunikacji sieciowej. Zaprezentujemy też ograniczenia warstwowej architektury TCP/IP w komunikacji z terminalami ruchomymi, szczególnie w kontekście nowych usług i specjalizowanych aplikacji. W dalszej części rozdziału skupimy się na przeglądzie najważniejszych koncepcji i protokołów obsługi mobilności w warstwie sieciowej - rozwiązań uniwersalnych i dopracowanych w stopniu pozwalającym na ich wdrożenie już w chwili obecnej, w aktualnie wykorzystywanych systemach sieciowych. Druga część rozdziału poświęcona będzie z kolei radykalnie nowym roz-

wiązaniom protokolarnym (wybiegającym poza klasyczne koncepcje Internetu) i projektom mającym u swych podstawowych założeń konieczność zaadresowania potrzeb rosnącej rzeszy użytkowników ruchomych.

## 2 Charakterystyka i wymagania związane z realizacją usług w sieciach IP

W ostatnich latach jesteśmy świadkami głębokich zmian na rynku usług telekomunikacyjnych. Pojawienie sie nowych technologii sieciowych oraz wdrożenie szeregu nowych systemów otworzyło możliwość praktycznej realizacji bardzo szerokiego wachlarza usług dostępnych dla użytkowników końcowych. Dodatkowo bezprecedensowy rozwój i popularyzacja sieci teleinformatycznych posunęły się na tyle daleko, iż świadczenie usług z ich użyciem stało się koniecznością dla usługodawców z branż nie łączonych wcześniej z wykorzystaniem sieci teleinformatycznych, jak np. firmy wydawnicze klasyczne i audio/wideo, administracja publiczna, dostawcy energii, banki itp.

Co więcej, oczekiwania użytkowników dotyczą możliwości korzystania z powyższych usług nie tylko za pośrednictwem urządzeń stacjonarnych ale, i to przede wszystkim, terminali mobilnych. Powstaje pytanie, czy dostępne obecnie systemy są w stanie zapewnić wystarczającą do ich realizacji jakoś transmisji danych, szczególnie wobec faktu, iż wymagania danej usługi pozostają najczęściej takie same zarówno w przypadku użytkownika stacjonarnego jak i mobilnego.

W Tabeli 1 zaprezentowano wymagania stawiane aplikacjom zaliczanym do poszczególnych klas QoS (*Quality of Service*) wg zaleceń ITU-T (*Internetional Telecommunication Union - Telecommunication Standardization Sector*). Parametry, których pożądane wartości zostały określone z zaleceniu ITU-T Y.1541 [4] to:

- IPTD (IP Packet Transfer Delay) - opóźnienie transmisji pakietów,
- IPDV (IP Packet Delay Variation) - zmienność opóźnienia (często nazywana jitterem),
- IPLR (IP Packet Loss Ratio) - współczynnik utraty pakietów,
- IPER (IP Packet Error Ratio) - współczynnik błędnie przesłanych pakietów.

O ile w przypadku systemów przewodowych, zachowanie pożądanych parametrów transmisji nie przedstawia dużego problemu (zakładając, że dany system nie jest przeciążony), to w systemach bezprzewodowych (najatrakcyjniejszych jako systemy dostępowe z punktu widzenia użytkownika mobilnego) sprawa jest znacznie trudniejsza.

Przykładowo, w zleceniu ITU-T G.114 [5] przedstawiono wartości opóźnień transmisji charakterystyczne dla różnych technik jej realizacji. Zaprezentowane wartości wskazują na wyraźną różnica między opóźnieniem wprowadzonym przez sieci przewodowe, a tym charakteryzującym transmisję bezprzewodową. W przypadku tych ostatnich, pożądane (dla realizacji interaktywnych usług czasu rzeczywistego) opóźnienie na poziomie zbliżonym do 40ms udaje się uzyskać [6] dopiero dla systemów LTE (*Long Term Evolution*) [7].

| Klasa usługi | Przykładowe aplikacje | IPTD | IPDV | IPLR | IPER |
|---|---|---|---|---|---|
| **Klasa 0** | Aplikacje czasu rzeczywistego, wrażliwe na zmienność opóźnienia, cechujące się wysoka interaktywnością (gry online, konferencje audio i wideo) | 100 ms | 50 ms | $10^{-3}$ | $10^{-4}$ |
| **Klasa 1** | Aplikacje czasu rzeczywistego, wrażliwe na zmienność opóźnienia, interaktywne (VoIP, konferencje audio i wideo) | 400 ms | 50 ms | $10^{-3}$ | $10^{-3}$ |
| **Klasa 2** | Dane transakcyjne, cechujące się wysoka interaktywnością (sygnalizacja) | 100 ms | N | $10^{-3}$ | $10^{-3}$ |
| **Klasa 3** | Dane transakcyjne | 400 ms | N | $10^{-3}$ | $10^{-3}$ |
| **Klasa 4** | Aplikacje wymagające niewielkich strat pakietów (strumieniowanie multimediów, telewizja na żądanie) | 1 s | N | $10^{-3}$ | $10^{-3}$ |
| **Klasa 5** | Tradycyjne aplikacje | N | N | N | N |

**Tablica 1.** Wymagania stawiane aplikacjom zaliczanym do poszczególnych klas (N - nie wyspecyfikowano) [4]

Ponadto, w systemach w których mobilność terminali wymusza realizację procedur ich przełączenia pomiędzy fizycznymi punktami dostępu do sieci, obserwujemy dodatkowe straty pakietów, opóźnienia w transmisji oraz zmienność tego opóźnienia. Efekty te związane są z chwilowym przerwami w komunikacji terminala z infrastrukturą sieci dostępowej w czasie trwania procesu przełączenia, którego czas trwania przełączenia jest zależny od wykorzystywanego scenariusza mobilności, w techniki dostępowej, a nawet sterowników sprzętu zlokalizowanych w terminalu.

Nawet jeżeli przerwy w transmisji związane z przełączaniem oraz realizacją procedur mobilności w warstwie IP są niewielkie, mogą one niekorzystanie wpływać na jakość usług - zwłaszcza usług strumieniowania audio lub wideo oraz usług konwersacyjnych (rozmowa, wideo rozmowa, konferencja). Wysokie wymagania, zyskujących coraz większą popularność interaktywnych gier online, powodują, że realizacja przełączenia użytkowników jest sporym wyzwaniem, a każda jego optymalizacja przynosi znaczące korzyści.

Należy podkreślić fakt, że czas przełączania ma znaczenie nie tylko w przypadku usług czasu rzeczywistego, lecz zbyt długi może skutkować zerwaniem nawet połączeń transmisji danych (np. w przypadku korzystania z protokołów połączeniowych typu TCP).

### 2.1 Ograniczenia w komunikacji z obiektami ruchomymi w ujęciu architektury ISO-OSI

Jak wspomniano wcześniej wykorzystywane aktualnie architektury warstwowe, zarówno TCP/IP jak też traktowana jako model odniesienia architektura ISO-OSI projektowane były z myślą o użytkownikach i urządzeniach stacjonarnych.

| **Warstwa** | **Wpływ poruszania się terminali** |
|---|---|
| Warstwa fizyczna | Zmienność jakości sygnału w łączu radiowym. |
| Warstwa łącza danych | Zmienność jakości sygnału łącza dostępowego; możliwość utraty pakietów, przepełnienia kolejek. |
| Warstwa międzysieciowa | Możliwość zmiany parametrów sieciowych (w tym adresu IP i routingu) wprowadzające potencjalne straty pakietów, opóźnienia oraz utratę możliwości identyfikacji węzła mobilnego. |
| Warstwa transportowa | Możliwość zerwania połączenia lub znaczącego pogorszenia parametrów jakościowych transmisji. |
| Warstwa aplikacji | Konieczność śledzenia zmiany konfiguracji warstw niższych (np. zmiany adresu IP) przez aplikacje świadome mobilności w celu modyfikacji parametrów ich działania. |

**Tablica 2.** Wpływ poruszania się terminala na poszczególne warstwy architektury TCP/IP

W efekcie mechanizmy praktycznie każdej z warstw powyższego modelu wprowadzają znaczące ograniczenia w efektywnej obsłudze użytkowników ruchomych (patrz Tabela 2) [8], [9].

Sytuację komplikuje też założenie od możliwie daleko posuniętej separacji pomiędzy poszczególnymi warstwami - rozwiązaniu sprawdzonym w przypadku sieci o statycznej strukturze, lecz bardzo kłopotliwym w przypadku konieczności obsługi elementów mobilnych.

**Ograniczenia warstwy łącza danych**

Szybka zmiana fizycznego punktu podłączenia do sieci, charakteryzująca się możliwie niewielkim zakłóceniem parametrów jakościowych łączności pomiędzy terminalem ruchomym, a siecią dostępową, stanowi podstawę efektywnej obsługi tego rodzaju użytkownika. Wszelkie zakłócenia łączności w tej warstwie będą miały wpływ na efektywność funkcjonowania warstw wyższych.

Większość dostępnych na rynku technik bezprzewodowych sieci dostępowych oferuje możliwość wsparcia mobilności w postaci dodatkowych mechanizmów, bardzo często specyficznych dla konkretnego producenta i niepodlegających standaryzacji. Rozwiązania te w znacznej mierze ograniczają się jedynie do sieci homogenicznych i zapewniają funkcjonalności związane z mobilnością wyłącznie w warstwie łącza danych. Zazwyczaj, właśnie heterogeniczny charakter sieci uniemożliwia zapewnienie mobilność w warstwie łącza danych i przenosi ten wymóg na wyższe warstwy.

W celu szybkiego i bezstratnego przełączenia istnieje potrzeba zapewnienia odpowiednich procedur w warstwach wyższych, w szczególności na poziomie warstwy sieciowej modelu ISO-OSI, czy też tzw. międzysieciowej w TCP/IP. Szczegóły tych mechanizmów zależą od typu i zasięgu mobilności, lecz separacja międzywarstwowa pomiędzy warstwą łącza danych, a warstwami wyższymi powoduje znaczące utrudnienie w ich skutecznym działaniu.

**Ograniczenia warstwy sieciowej**

Adresacja IP w tradycyjnym modelu TCP/IP odgrywa rolę zarówno identyfikatora danego urządzenia (ID), jak również opisuje jego lokalizację w strukturze sieci (LOC). Tego rodzaju sztywne powiązanie prowadzi do szeregu komplikacji, gdyż w trakcie przełączania, w związku z zmianą punktu podłączenia węzła mobilnego MH (*Mobile Host*) do sieci, jego adres IP musi ulec zmianie. Urządzenie otrzymuje nowy adres zgodny z adresacją obowiązującą w danej sieci dostępowej, odzwierciedlający zmianę jego lokalizacji w strukturze systemu sieciowego.

Niestety, zmiana adresu powoduje zarazem zmianę jedynego dostępnego w sieci IP identyfikatora danego terminala, co uniemożliwia zachowanie ciągłości komunikacji z węzłami korespondującymi CN (*Corresponding Nodes*). Nawet jeżeli terminal uzyska nowy adres automatycznie komunikacja z nim zostanie przerwana. Co więcej, bez dodatkowych procedur węzeł ten nie będzie już osiągalny dla węzłów korespondujących, ze względu na niemożność ustalenia przez nie jego nowego identyfikatora.

**Ograniczenia warstwy transportowej**

W architekturze TCP/IP protokoły warstwy transportowej oferują połączenia end-to-end. W przypadku protokołu TCP mówimy o przekazie zorientowanym połączeniowo, gwarantującym niezawodny i integralny przekaz pakietów, pozwalający na sterowanie przepływem i przeciwdziałanie przeciążeniom w sieci. Przy zestawianiu sesji połączeniowych TCP wykorzystywane są parametry konfiguracyjne warstwy sieciowej (oferującej usługi warstwie wyższej), w tym adresy IP, co prowadzi do problemów opisanych powyżej.

Dodatkowo, nawet jeśli zastosowane rozwiązania obsługi mobilności pozwalają na zachowanie stałego adresu również po przełączeniu, to zbyt długi czas jego trwania spowoduje zadziałanie mechanizmy time-out, które również doprowadzą do zerwania połączenia.

**Ograniczenia warstwy aplikacji**

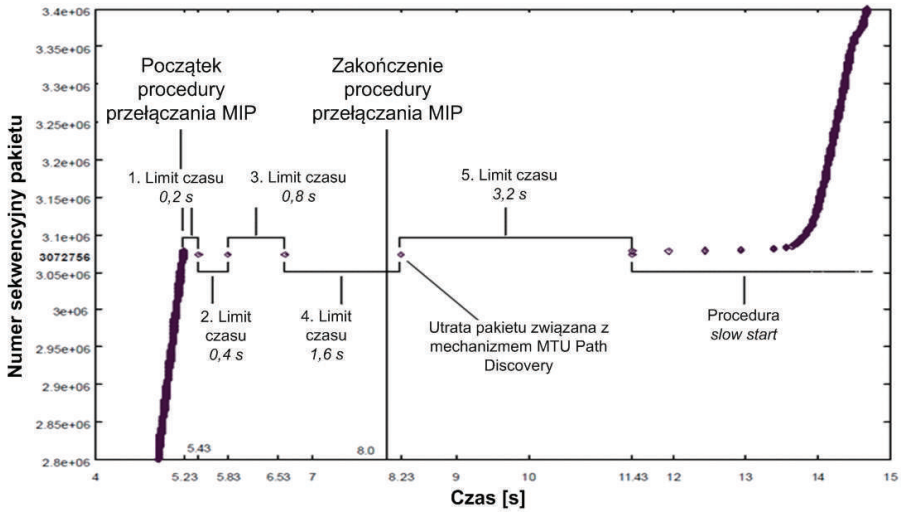
Biorąc pod uwagę fakt, iż obsługę mobilności wprowadzamy na potrzeby realizacji usług dla użytkownika końcowego realizowanych przez różnorodne aplikacje, należy zaznaczyć, iż pożądane są rozwiązania całkowicie uniezależniające aplikacje od typu obsługiwanego terminala. Sposób działania aplikacji nie powinien zależeć od tego, czy odbiorcą usługi jest terminal stacjonarny czy ruchomy.

Jednakże, wobec braku wdrożonych rozwiązań obsługi mobilności w warstwach niższych, wiele aplikacji zmodyfikowano w celu umożliwienia im obsługi użytkowników ruchomych. Stosowane rozwiązania mobilności w warstwie aplikacyjnej wykorzystują funkcje dedykowanego protokołu sygnalizacji (np. SIP - *Session Initiation Protocol*) pozwalające na zmianę parametrów (np. docelowego adresu IP) transmisji danych. Wymaga to jednak od aplikacji ciągłego monitorowania parametrów interfejsu sieciowego, co w związku z opisanym w poprzednim punkcie brakiem mechanizmów współpracy międzywarstwowej jest trudne.

Implementacja mobilności na poziomie aplikacji prowadzi do konieczności wdrożenia odpowiednich mechanizmów w każdej aplikacji niezależnie, co w oczywisty sposób zwiększa narzut protokolarny i ogranicza skalowalność.

**Brak współpracy międzywarstwowej**

Zgodnie z koncepcją TCP/IP protokoły warstwy transportowej korzystają z usług dostarczanych przez warstwę sieciową, nie uwzględniając mechanizmów warstw niższych, w tym współpracujących z warstwą fizyczną sieci bezprzewodowych. W związku z tym, utrata pakietów związana z przełączaniem traktowana jest tak samo jak utrata pakietów na skutek niskiej jakości lub przeciążenia łącza. Przykładowa ilustracja tego problemu została zaprezentowana na Rys. 2.




**Rysunek 2.** Opóźnienie w kontynuacji transmisji TCP po dokonaniu przełączenia, ze względu na brak mechanizmów współpracy międzywarstwowej (na podstawie [10]).

Jak widać, transmisja TCP zostaje przerwana w momencie rozpoczęcia przełączenia, które zostaje zakończone po około 3 s. Protokół TCP uznaje brak łączności za spowodowany niską jakością lub przeciążeniem łącza i próbuje retransmitować niepotwierdzone pakiety, wybierając stopniowo coraz dłuższe okresy pomiędzy tymi próbami. Po zakończeniu przełączenia, mechanizm MTU Path Discovery [11] powoduje jeszcze utratę pojedynczego pakietu. W efekcie przerwa w łączności wynosi około 6 s, z czego tylko 3 przypadają na rzeczywiste przełączenie w warstwach niższych. Dodatkowo obserwujemy działanie mechanizmu slow-start, poważnie ograniczającego szybkość wznowionej transmisji przez kilka kolejnych sekund.

Zastosowanie mechanizmów współpracy międzywarstwowej pomiędzy warstwą łącza danych (podwarstwą MAC) i warstwą transportową pozwoliłoby tu nie tylko na skrócenie czasu przełączenia, lecz także na wznowienie transmisji protokołu TCP natychmiast to po jego zakończeniu i z pełną szybkością.

### 2.2 Nowe wymagania środowiska mobilnego

Poza opisanymi powyżej parametrami QoS jakości transmisji danych, realizacja niektórych usług istotnych dla użytkowników mobilnych może też wymagać wprowadzenia całkowicie nowej funkcjonalności ze strony systemu sieciowego. Obok usług tradycyjnych, przy świadczeniu których, wystarczy spełnić rygorystyczne wymagania na parametry sieciowe, możemy wymienić szereg dodatkowych usług, atrakcyjnych i ważnych dla przemieszczających się terminali. Standardowa transmisja danych nawet tych multimedialnych staje się dla użytkownika niewystarczająca. Mnogość dostępnych informacji i usług powoduje, że kluczowa z jego puntu widzenia, staje się łatwość w odnalezieniu opowiadających mu usług. Duże znaczenie w tego typu mechanizmach ma odpowiednie określenie właściwości obiektu. W przypadku użytkowników mobilnych, który w sposób naturalny zmieniają swoje położenie i punkt podłączenia do sieci, lokalizacja jest jedną z ważniejszych własności.

Dla usług krytycznych związanych z bezpieczeństwem np. z zintegrowanym systemem ratownictwa pożądana jest dokłada lokalizacja - nazywana często *geolokalizacją*. W ramach takiej usługi ustalana jest precyzyjna pozycja geograficzna terminala czy grupy terminali. W związku z możliwością podłączenia użytkownika do różnych sieci, stosowania tuneli i translacji adresów jej implementacja nie jest zagadnieniem trywialnym.

Niejednokrotnie, w przypadku aplikacji o charakterze np. informacyjnym, geolokalizacja dostarcza pozycji z nadmiarową dokładnością i prowadzi do dodatkowych obliczeń związanych z określeniem położenia względem innych obiektów. W przypadku takich aplikacji, pożądana jest informacja o lokalizacji w okolicy konkretnego obiektu np. zabytku, czy pojazdu komunikacji miejskiej. Możemy w takim przypadku mówić o *lokalizacji kontekstowej*, która pozwoli na udostępnienie informacji o położeniu terminala a kontekście innego obiektu (np. turystycznego), usługi (np. gastronomicznej), czy grupy użytkowników (np. uczestników tej samej wycieczki). Pozwala to na świadczenie usług przeznaczonych do grona dedykowanych odbiorców. Przykładowo użytkownikom terminali mobilnych w okolicy konkretnego zabytku może zostać wyświetlona szczegółowa informacja, w tym treści multimedialne dotyczące tego obiektu, w pojazdach komunikacji miejskiej wyświetlana może być informacja o aktualnym, rozkładzie jazdy z rzeczywistymi czasami przejazdu i sugestiami przesiadek.

## 3 Podstawowe wymagania funkcjonalne dla mobilnego Internetu

W naszych dalszych rozważaniach mobilność Internetu traktować będziemy jako zdolność sieci do realizacji nieprzerwanej komunikacji przez urządzenie ruchome

MH, podczas jego przemieszczania się miedzy różnymi sieciami (podsieciami) IP. Pominiemy dalej przypadki wspierania mobilności terminali w warstwie łącza danych (zależnej od zastosowanej techniki dostępu) - traktując taki przypadek, jako w pełni rozwiązywany przez odpowiednie mechanizmy tej warstwy.

By umożliwić realizację nowoczesnych usług i aplikacji oferowanych urządzeniom mobilnym na poziomie analogicznym do realizowanego przez urządzenia stacjonarne, musimy rozwiązać szereg problemów i ograniczeń, wynikających chociażby z niedopasowania klasycznej architektury Internetu do potrzeb terminali ruchomych. Zestaw powyższych wymagań i istotnych problemów obejmuje między innymi [9]:

**Mechanizmy zarządzania przełączeniami (*handover management*)**

Konieczne jest opracowanie nowych procedur pozwalających na realizację nieprzerwanej komunikacji między węzłem korespondującym CN, a urządzeniem ruchomym MH podczas przemieszczania się tego urządzenia i zmiany jego punktu przyłączenia do sieci. Głównym celem mechanizmów zarządzania przełączaniem (tzw. mechanizmów rejestracji) będzie minimalizacja "zakłóceń" - przerw i związanych z nimi strat pakietów - w komunikacji podczas procesu przełączania.

**Zarządzanie lokalizacją (*location management*)**

Niezbędne jest również posiadanie aktualnej informacji o lokalizacji MH. W środowisku sieci IP adres stacji pełni zarówno rolę jej unikalnego identyfikatora (ID - *identifier*), jak i parametru pozwalającego ją zlokalizować w strukturze sieci (LOC - *locator*). W środowisku mobilnym te dwie wartości z założenia powinny być rozróżnialne. Jest to związane z często występującą zmianą punktu przyłączenia (tj. zmianą domeny, czy sieci IP), co prowadzi do konieczności zmiany parametru LOC, podczas gdy identyfikator (ID) powinien być zachowany.

**Multihoming**

By poprawić parametry jakościowe i wydajnościowe sieci i/lub urządzenia końcowego, np. zwiększyć przepływ pakietów, pożądana jest implementacja funkcjonalności multihomingu. Wiąże się ona z wyposażeniem MH w szereg interfejsów sieciowych, umożliwiających wybór jednej z wielu ścieżek połączeniowych z wykorzystaniem tej samej, bądź różnych sieci dostępowych. Pozwala to, w szczególności na wybór rozwiązania spełniającego wymagania jakościowe, względnie optymalizującego inną funkcję kryterialną.

Jest to tym ważniejsze, iż wykorzystanie pojedynczej techniki komunikacyjnej nakłada na użytkownika terminala ruchomego dość istotne ograniczenia. Po pierwsze, technika komunikacyjna sama w sobie posiada charakterystyczne dla siebie ograniczenia np. w postaci nominalnej przepustowości. Zwykle z czasem wartości te stają się niewystarczające i wprowadzana jest nowa technika. Prowadzi to do stworzenia heterogenicznego środowiska, w którym wcześniej mobilność nie występowała, lub była ograniczona do pojedynczej techniki komunikacyjnej.

Istotną kwestią staje się zatem wypracowanie innych koncepcyjnie rozwiązań, które pozwalałyby spajać także środowiska heterogeniczne. Taką koncepcją jest realizacja mobilności w warstwie modelu OSI, zlokalizowanej powyżej wszystkich

mechanizmów techniki dostępowej. Koncepcyjnie rozwiązaniem takim właśnie rozwiązaniem jest mobilność realizowana z wykorzystaniem protokołu IP.

**Aplikacje**

Sieć oferująca obsługę mobilności powinna wspierać zarówno aplikacje wykorzystywane obecnie przez użytkowników stacjonarnych, jak i te nowo wprowadzane, w tym przeznaczone specyficznie dla użytkowników mobilnych. Te ostatnie mogą wymagać od sieci dodatkowej funkcjonalności, jak na przykład wspomniane już usługi lokalizacyjne.

W kontekście pierwszej ze wspomnianych grup aplikacji, ważne jest by efektywne działanie w środowisku mobilnym nie wymagało z ich strony żadnych modyfikacji.

**Bezpieczeństwo**

Jednym z najważniejszych wyzwań dla współczesnego Internetu jest jego bezpieczeństwo. Wraz z wprowadzaniem nowych mechanizmów kontroli i zarządzania elementami sieci, do których zaliczają się również mechanizmy obsługi mobilności, rośnie również ryzyko ataku. Biorąc pod uwagę, iż mechanizmy sieciowe związane z zarządzaniem mobilnością, obejmują np. procedury uwierzytelniania użytkowników i urządzeń, pozyskiwania adresów, czy rejestracji, powinny uwzględniać właściwe zabezpieczenia.

Powyższa lista nie wyczerpuje oczywiście wszystkich elementów pożądanych dla efektywnej obsługi terminali ruchomych. Kolejne postulaty wymieniane w [12] dotyczą sposobu zapewnienia wsparcia mobilności. Modelowo, mobilność, ze względu na rosnącą popularność terminali przenośnych, powinna być mechanizmem wbudowanym w rozwiązanie sieciowe, a nie funkcjonalnością opcjonalną stanowiącą rozszerzenie podstawowej. Wiąże się to z kwestią nieoptymalnego rozwiązywania problemów powstałych na skutek zaprojektowania architektury bez naturalnych założeń odnośnie obsługi mobilności. Ponadto, transmisja ruchu na potrzeby terminali mobilnych powinna być realizowana w postaci rozproszonej. Pozwala to rozwiązać kwestię skalowalności rozwiązania, dość silnie obserwowaną w szeregu obecnie wykorzystywanych rozwiązań. Jako dodatkowe, postulowane jest odseparowanie płaszczyzny sterowania i transmisji danych. Dalsze postulaty dotyczą możliwości wykorzystania wielu różnych protokołów, zamiast jednego uniwersalnego. Jest to także związane z dwoma proponowanymi modelami obsługi terminali ruchomych - end-to-end i realizowanych po stronie sieci. Wreszcie - sposób adresacji węzłów w Internecie Przyszłości powinien wyróżniać się rozwiązaniami kwestii skalowalności i mobilności już w swojej architekturze.

## 4 Przegląd protokołów mobilności

Na przestrzeni ostatnich 20 lat zaproponowano kilkadziesiąt różnych rozwiązań protokolarnych do obsługi mobilnych użytkowników w sieci Internet. Kilkanaście z nich zostało opracowanych jako standardy IETF w postaci dokumentów RFC. Reprezentatywną (choć daleką od pełnej) listę tych protokołów zawiera informacyjny dokument RFC 6301 [13] Przedstawione w nim oraz w literaturze

przedmiotu rozwiązania charakteryzują się specyficznymi funkcjonalnościami pozwalającymi na optymalizację wybranych problemów sieciowych napotykanych przy obsłudze użytkowników mobilnych w sieciach IP, bądź zastosowanie protokołu mobilności w określonym kontekście, tj. do realizacji ściśle określonych usług i aplikacji sieciowych.

Można zaobserwować, iż większość "klasycznych" rozwiązań zarządzania mobilnością, włączając w to najpopularniejsze protokoły z grupy Mobile IPv4 i v6, wykorzystuje dedykowany komponent sieciowy, zwany agentem mobilności. Głównym jego zadaniem jest zapewnienie identyfikacji i lokalizacji stacji ruchomej z wykorzystaniem pary logicznie powiązanych adresów IP.

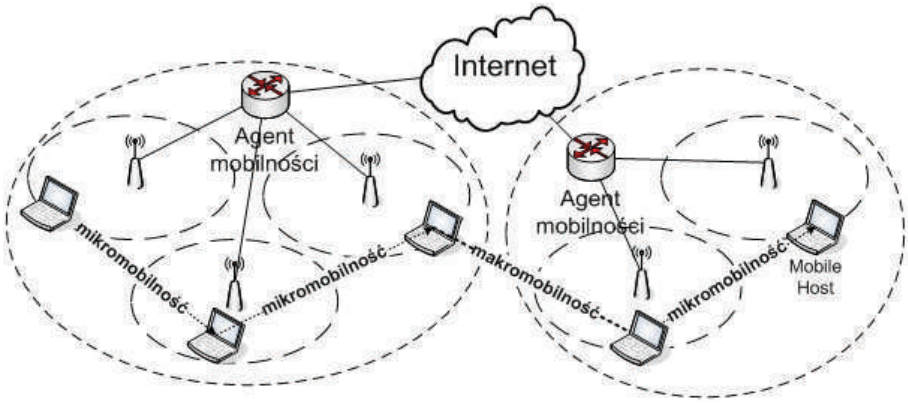
Tego rodzaju podstawowe koncepcje są, szczególnie w ostatnim okresie, rozbudowywane o nowe funkcjonalności pozwalające na dynamiczne zarządzanie agentami mobilności. Istnieje też grupa rozwiązań zalecających wykorzystanie identyfikatorów abstrakcyjnych (nieposiadających formy adresu sieciowego), a nawet opisowych nazw urządzeń. Pozwala to na uwolnienie się od ograniczeń związanych z bezpośrednim stosowaniem adresacji IP, poprzez odwzorowywanie tego rodzaju nazw na aktualne adresy IP terminala mobilne, z wykorzystaniem dynamicznie modyfikowanych zapisów w bazach serwerów DNS. Te nowe zagadnienia będą przedmiotem naszych zainteresowań w drugiej części rozdziału.

Jedną z najczęściej proponowanych klasyfikacji protokołów mobilności jest ich podział ze względu na jej skale przemieszczeń i zakres realizowanych przełączeń. W tym kontekście mówimy zwykle o mobilności globalnej, związanej z przemieszczaniem się między domenami (*inter-domain mobility*), bądź o przemieszczaniu się w "mniejszej", lokalnej skali, tj. wewnątrz określonej domeny (*intra-domain mobolity*). Ilustrację obu scenariuszy przemieszczeń i związanych z nimi przełączeń zawarto na Rys. 3. Domena definiowana jest przy tym jako sieć lub fragment większej sieci podlegający jednolitemu zarządzaniu. Scenariusz ten nazywamy również mikromobilnością.

Mobilność międzydomenowa, nazywana zwykle makromobilnością, odnosi się do przełączeń między niezależnymi pod względem zarządzania systemami sieciowymi. Rozwiązania tego typu mają za zadanie zapewnić globalne zarządzanie mobilnością w całej intersieci, niezależnie od systemu dostępowego wykorzystywanego aktualnie przez dany terminal mobilny. W związku z tym mechanizmy stosowane w tych protokołach się znacznie bardziej złożone - zawierając takie elementy jak: uwierzytelnianie, przydzielanie i weryfikację adresu IP, ich rejestrację oraz aktualizację całej ścieżki przesyłania danych między CN i MH.

Ponieważ scenariusz, w którym użytkownik przełącza się w ramach jednej domeny administracyjnej (mobilność wewnątrzdomenowa - mikromobilność) występuje częściej, zaproponowane zostały mechanizmy "poświęcające" uniwersalność rozwiązania na rzecz uzyskania poprawy efektywności działania w obrębie domeny administracyjnej. Związane z tym ograniczenie wymagań na procedury rejestracji i modyfikacji trasy, pozwala na uzyskanie znacznych korzyści, w postaci skrócenia czasu przełączania oraz poprawy efektywności wykorzystania zasobów sieciowych (między poprzez ograniczenie ruchu sygnalizacyjnego).

Należy zwrócić uwagę, iż w wielu przypadkach rozwiązania makro- i mikromobilności mogą funkcjonować równolegle, łącząc globalną obsługę mobilności ze zwiększoną efektywnością działania w obrębie pojedynczych domen administracyjnych.




**Rysunek 3.** Porównanie mikromobilności i makromobilności.

Innym, często spotykanym, podziałem rozwiązań mobilności, jest klasyfikacja wg miejsca ulokowania elementów wspierających mobilność (*mobility aware entities*) [14].

W przypadku modelu terminal - względnie host-based mobility (HBM) szereg funkcjonalności implementowanych jest w węźle klienckim stacji mobilnej (*Mobile Host - MH*). W takim przypadku bezprzewodowe łącze radiowe wykorzystywane jest zarówno do przesyłana danych, jak i sygnalizacji. Z kolei, w modelu *operator-* czy też *network-based mobility* (NBM) [14], wszystkie elementy systemu wspierające mobilność zlokalizowane są po stronie sieci, co z reguły oznacza konieczność wprowadzenia dodatkowego elementu infrastruktury sieciowej wykorzystywanego do realizacji sygnalizacji w imieniu terminala mobilnego.

Poniżej przedstawiamy przykłady dwóch najpopularniejszych i intensywnie rozwijanych rozwiązań obsługi mobilności w sieciach IP. Mobile IP jest przykładem globalnego rozwiązania typu HBM, podczas gdy Proxy Mobile IP należy do grupy rozwiązań mikromobilności, zrealizowanych zgodnie z modelem NBM. Oba rozwiązania przedstawiono tu w ich wersjach podstawowych (tj. bez zaawansowanych optymalizacji). Należy pamiętać, iż każde z nich doczekało się licznych modyfikacji i rozszerzeń, których wybrane przykłady przedstawimy w drugiej części rozdziału.

### 4.1 Mobile IP

Mobile IP (MIP) [15], [16], [17], [18], [19], [20] jest niewątpliwie najpopularniejszym protokołem mobilności proponowanym dla sieci IP. Dzięki realizacji usługi mobilności w warstwie sieciowej w MIP zapewnia się przezroczystość przełączenia dla warstw wyższych, a jednocześnie jest w stanie funkcjonować niezależnie od wykorzystywanej przez urządzenie mobilne techniki dostępu do sieci. W dokumentach IETF wyszczególnia się dwie grupy zaleceń związanych z MIP - te związane z obsługą mobilności w sieci wykorzystującej IPv4 oraz IPv6.

- Węzeł ruchomy (*Mobile Host - MH*),
- Węzeł korespondujący (*Correpsonding Node - CN*),
- Agent domowy (*Home Agent - HA*),
- oraz wyłącznie w przypadku sieci IPv4: Agent obcy (*Foreign Agent - FA*).

Podstawą funkcjonowania protokołu MIP jest wykorzystanie dwóch adresów węzła mobilnego - jednego powiązanego z jego siecią macierzystą, nazywanego adresem domowym (*home address*) oraz drugiego, zgodnego z adresacją sieci aktualnie odwiedzaną przez terminal mobilny, nazywanego adresem obcym/tymczasowym (*care-of address - CoA*). Adres domowy pozostaje niezmienny i stanowi stały identyfikator węzła mobilnego (ID). Z kolei CoA opisuje aktualną lokalizację węzła mobilnego w strukturze sieci - pełni zatem rolę lokalizatora (LOC).

Wynika stąd, iż po każdym przyłączeniu do nowej sieci terminal mobilny otrzymuje inny adres tymczasowy, który musi zostać powiązany ze stałym adresem domowym MH. Elementem odpowiedzialnym za zapewnienie aktualności takiego powiązania jest sam węzeł mobilny, który inicjuje tzw. procedurę rejestracji po każdorazowym wykryciu zmiany adresu CoA.

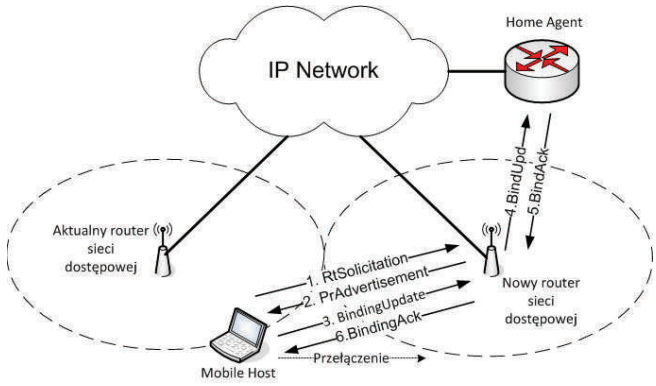
Z kolei elementem systemu MIP przechowującym powyższe powiązanie jest agent domowy HA. Tak więc w ramach procedury rejestracja informacja o nowym CoA musi trafić do niego. W zależności od wersji protokołu IP, węzeł mobilny albo kontaktuje się bezpośrednio ze swoim agentem domowym (IPv6) lub też z agentem obcym obsługującym sieć, w której się aktualnie znajduje (IPv4) i który odpowiada za przekazanie informacji rejestracyjnej do właściwego HA.

Można uznać, iż para adresów właściwa dla każdego węzła mobilnego niesie następujące informacje:

- adres domowy - identyfikuje: węzeł mobilny i lokalizuje jego sieć domową i agenta domowego,
- adres tymczasowy - identyfikuje i lokalizuje węzeł mobilny.

Węzły korespondujące CN, wysyłając pakiety IP do węzła mobilnego, posługują się jego adresem domowym (identyfikatorem). Jak wskazano powyżej, ruch taki trafia jednak do sieci domowej węzła mobilnego, a nie do jego aktualnej lokalizacji. Agent domowy jest zatem zobowiązany do odebrania tego rodzaju ruchu i "przekierowania go" do aktualnej lokalizacji węzła ruchomego, opisanej jego aktualnym adresem CoA. Odbywa się to poprzez bezpośrednie tunelowanie

ruchu pod adres CoA (IPv6) lub też pośrednio, poprzez przekazanie ruchu tunelem do odpowiedniego agenta obcego, który odpowiada za przekazanie go do węzła docelowego.




**Rysunek 4.** Procedura przełączenia protokołu MIPv6.

Przykładowy scenariusz procedury przełączenia z wykorzystaniem protokołu Mobile IPv6 został zaprezentowany na Rys. 4. W przypadku przejścia terminala mobilnego do sieci innej niż jego domowa następuje wykrycie zmiany punktu podłączenia na podstawie wiadomości *Router Advertisement*. Wykonywana jest następnie procedura rejestracji do nowej sieci i powiązanie aktualnego tymczasowego adresu z adresem domowym.

Pomimo stosunkowo małej złożoności powyższego rozwiązania, pojawia się tu kilka problemów, wymagających uwagi. Pierwszym z nich jest szybkie wykrycie przełączenia przez terminal ruchomy, który powinien w możliwie najkrótszym czasie po zmianie sieci podłączenia przeprowadzić procedurę rejestracji, gdyż do tego czasu nie jest w stanie odbierać adresowanego do siebie ruchu. Bez wprowadzenia elementów współpracy międzywarstwowej, jest to zadanie trudne do efektywnej realizacji.

Kolejnym problemem jest tzw. routing trójkątny, wynikający z konieczności przesyłania danych adresowanych do MH do jego sieci domowej, skąd są dopiero dostarczane do jego aktualnej lokalizacji, podczas gdy ruch z MH do CN wysyłany jest bezpośrednio. Rozwiązanie takie, w połączeniu z narzutem wprowadzanym przez mechanizmy tunelowania, nie wpływa dobrze na efektywność wykorzystania sieci.

Problemy te stały się powodem opracowania szeregu rozszerzeń i optymalizacji podstawowego protokołu MIP, które przedstawimy w drugiej części rozdziału.

Podsumowując, protokół MIP jest w stanie zapewnić węzłom ruchomym podstawową obsługę mobilności w skali globalnej. Jedną z jego podstawowych zalet jest możliwość wdrożenia tego protokołu, bez konieczności znaczącej modyfika-

cji istniejącej infrastruktury sieciowej. W przypadku protokołu MIPv6 wystarczy nam ulokowanie HA w sieci domowej węzła mobilnego, oraz zaimplementowanie elementów systemu w samym MH, zainteresowanym uzyskaniem obsługi mobilności. Zarówno rdzeń sieci, jak i systemy odwiedzane przez MH nie podlegają modyfikacjom. Oczywiście zaleta powyższa dotyczy wersji protokołu MIPv6 i nie dotyczy MIPv4, czyniąc ją rozwiązaniem słabo dostosowanym do potrzeb powszechnej obsługi mobilności.

Należy tu podkreślić, iż ulokowanie elementów systemu w terminalu użytkownika, nie stanowi już tak dużego problemu jak w latach ubiegłych. Wprowadzenie uniwersalnych systemów operacyjnych przeznaczonych dla tego rodzaju urządzeń spowodowało, iż przestaliśmy mieć do czynienia z setkami modeli terminali klienckich, których odmienne środowiska programistyczne należało uwzględnić i oprogramować. Obecnie znacząca większość urządzeń wykorzystuje jeden z około 5 systemów operacyjnych. Jednocześnie rozwój i zróżnicowanie infrastruktury sieciowej po stronie operatorów sieci dostępowych, czyni wszelkie czynione tam modyfikacje przedsięwzięciem trudnym i kosztownym.

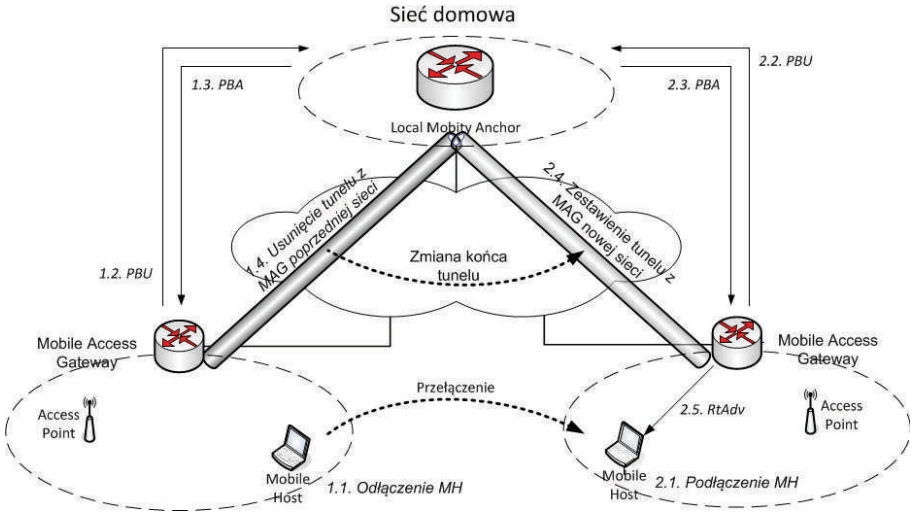
### 4.2 Proxy Mobile IPv6

Protokół Proxy Mobile IPv6 (PMIPv6) [21] jest rozwinięciem idei wykorzystanej w rozwiązaniu Mobile IPv6. PMIPv6 nie nakłada jednakże, w przeciwieństwie do MIPv6, wymagań związanych z implementacją mechanizmów mobilności i tym samym modyfikacją obsługi stosu sieciowego po stronie użytkownika. Jest to możliwe dzięki zastosowaniu podejścia, w którym cała procedura zapewnienia mobilności realizowana jest przez elementy infrastruktury sieciowej. Prowadzi do uzyskania kilku korzyści, w tym uniknięcia narzutu tunelowania oraz sygnalizacji na bezprzewodowym łączu dostępowym. Niestety, protokół Proxy Mobile IPv6 nie może być stosowany jako globalne rozwiązanie obsługi węzłów ruchomych, jako że należy on do grupy przeznaczonej do obsługi mikromobilności, czyli mobilności w obrębie określonej domeny administracyjnej. Proxy Mobile IPv6 wykorzystuje dwa dodatkowe elementy w infrastrukturze sieciowej, pokazane na Rys. 5. Są to:

- Mobile Access Gateway (MAG) oraz
- Local Mobility Anchor (LMA).

Oba te elementy zlokalizowane są po stronie infrastruktury operatora sieci dostępowej i zwalniają użytkownika z konieczności umieszczenia elementów systemu mobilności w jego terminalu mobilnym.

Element LMA pełni, w przypadku PMIP, rolę podobną do agenta domowego protokołu MIP. Jest on odpowiedzialny za śledzenie punktu podłączenia terminala ruchomego (co czyni dzięki wymianie odpowiednich wiadomości sygnalizacyjnych z elementami MAG), zestawienie tunelu do elementu MAG aktualnie obsługującego określony terminal mobilny i skierowanie ruchu do MH z jego użyciem.




**Rysunek 5.** Procedura przełączenia protokołu MIPv6.

Wobec braku, w rozwiązaniu PMIP, elementu systemu zlokalizowanego po stronie terminala ruchomego, uzyskujemy możliwość udostępnienia obsługi mobilności również terminalom starszym (*legacy terminals*), w przypadku których wdrożenie rozwiązania typu host-based (takiego jak MIP) byłoby bardzo kłopotliwe.

W tej sytuacji jednak, elementy zlokalizowane po stronie sieci muszą przejąć zadania realizowane uprzednio przez elementy umieszczone w terminalu użytkownika. W przypadku rozwiązania PMIP, zadania te realizowane są przez elementy MAG. Do tych zadań należą:

- wykrywanie podłączenia terminala,
- zestawienie i utrzymywanie tunelowanej komunikacji z elementem LMA,
- przenoszenie ruchu użytkownika pomiędzy terminalem klienckim i LMA.

W celu ich realizacji MAG musi zostać zlokalizowany w bezpośredniej bliskości podłączającego się terminala ruchomego, jako pierwsze, sąsiadujące z nim w strukturze sieci, urządzenie warstwy sieciowej.

Protokół PMIPv6 może zostać wykorzystany zarówno dla bezstanowej jak i stanowej metody automatycznej konfiguracji adresów IPv6. W przypadku wykorzystania protokołów DHCPv6 (*Dynamic Host Configuration Protocol for IPv6*) elementy MAG powinny pełnić rolę pośredników DHCP. W przypadku metody bezstanowej elementy te wysyłają wiadomości *Router Advertisement* przy pomocy transmisji unicast bezpośrednio do każdego z terminali mobilnych.

Niezależnie od wybranej metody przydzielana adresów, w PMIPv6 zastosowano podejście per-MN-prefix, w którym dla poszczególnych terminali mobilnych przydzielane są całe prefiksy IPv6, a nie pojedyncze adresy. Jest to z jednej

strony spora rozrzutność, która powoduje znaczące zmniejszenie dostępnej przestrzeni adresowej IPv6. Z drugiej strony taki zabieg wprowadza, w porównaniu do stosowanego w MIPv6 podejścia shared-prefix, możliwość ominięcia czasochłonnej procedury wykrywania duplikacji adresów.

Podsumowując, rozwiązanie PMIP należy do grupy rozwiązań typu network-based, nie wymagając żadnej modyfikacji terminala klienckiego. Jest to potencjalnie duża zaleta, umożliwiająca wprowadzenie obsługi mobilności na potrzeby użytkowników ruchomych, baz żadnego udziału z ich strony. Dodatkowo, każdy terminal zdolny do pracy w sieci IP jest w stanie skorzystać z wprowadzonej funkcjonalności, w przeciwieństwie do rozwiązania typu host-based, gdzie należy zapewnić implementację mechanizmów obsługi mobilności kompatybilną z określoną grupą urządzeń klienckich. Jednakże w świetle wspomnianych już zmian w oprogramowaniu terminali tego typu, zaleta ta nie jest już tak atrakcyjna, jak w przeszłości. Kolejną zaletą rozwiązania PMIP jest daleko posunięta kontrola mechanizmów zarządzania siecią nad procesem obsługi mobilności, co pozwala na łatwe wprowadzenie optymalizacji w wykorzystaniu zasobów sieciowych.

Należy też zauważyć, iż w przypadku rozwiązań host-based, bezprzewodowe łącze radiowe wykorzystywane jest zarówno do przesyłana danych, jak i sygnalizacji. Dodatkowo dane przesyłane są przez tunele, co ogranicza i tak przeważnie niewielką przepustowość łącza radiowego. Dla porównania, zastosowanie w rozwiązaniu PMIP modelu network-based przenosi te zadania na dodatkowy element infrastruktury sieciowej działający w imieniu terminala mobilnego, co pozwala uniknąć tego nadmiarowego narzutu.

Niestety, jak już wspomniano, PMIP nie jest rozwiązaniem makromobilności, a zatem nie umożliwi nam łatwego wprowadzenia jej powszechnej obsługi w sieci IP. W tym celu musiałoby ono zostać wdrożone u każdego z operatorów sieci dostępowej, a także należałoby zadbać o mechanizmy komunikacji pomiędzy utworzonymi w ten sposób domenami mikromobilności, w celu przekazywania informacji o aktualnej lokalizacji terminala ruchomego.

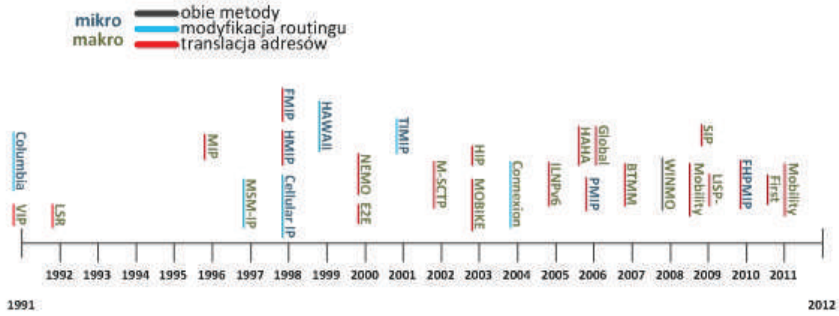
## 5 Modyfikacje i optymalizacje mechanizmów mobilności

W dotychczasowych rozdziałach dokonano klasyfikacji protokołów mobilności, koncentrując się na dwóch obszarach zagadnień. Pierwszy podział brał pod uwagę skalę dokonywanych przełączeń i wielkość sieci, w ramach której przemieszczają się urządzenia, czyli mobilności w skali globalnej bądź lokalnej, nazywanej odpowiednio makro- i mikromobilnością. Drugi z zaproponowanych podziałów uwzględniał podejście client / network side, związane z odpowiedzialnością terminala bądź też elementów sieciowych za procedury zarządzania mobilnością. W tym kontekście zaprezentowano przykładowe protokoły mobilności Mobile IPv4 i Mobile IPv6 oraz Proxy Mobile IPv6, ilustrujące proponowane klasyfikacje. Są to najczęściej spotykane rozwiązania podlegające szerokiej standaryzacji.

Pomimo iż przedstawione rozwiązania realizują podstawową funkcjonalność konieczną do obsługi użytkowników ruchomych, nie ulega wątpliwości, że wyma-

gają one optymalizacji, celem poprawy efektywności działania i skalowalności, dla zapewnienia wszystkich przedstawionych wymogów. Powyższe modyfikacje i sposoby optymalizacji zostaną przedstawione i przedyskutowane w kolejnych podrozdziałach, w których opisane zostaną również sposoby realizacji mobilności w innych warstwach niż warstwa sieciowa modelu TCP/IP. Dodatkowo zaprezentowane zostaną nowatorskie rozwiązania mobilności, wprowadzające nowe idee, zrywające z tradycyjną i dobrze znaną koncepcją dwóch adresów i agentów mobilności.

Na rysunku 6 zaprezentowano zestawienie protokołów mobilności z ich podstawową klasyfikacją na protokoły mikro- i makromobilności oraz protokoły wykorzystujące dynamiczną modyfikację routingu lub różnorodne metody translacji adresów.




**Rysunek 6.** Rozwój protokołów mobilności na przestrzeni ostatnich 20 lat.

Można zauważyć, że rozwiązania popularne w ostatnich latach (a zwłaszcza w ostatnich 10 latach) wykorzystują w głównej mierze koncepcję odwzorowywania unikalnego i niezmiennego identyfikatora (charakterystycznego dla każdego rozwiązania) na lokalizator (najczęściej adres IP). Mniejszą popularnością cieszą się natomiast rozwiązania wykorzystujące dynamiczną zmianę routingu. Przyczyną mogą tu być znaczące ograniczenia skalowalności, występujące w takich przypadkach. Konieczność umieszczenia informacji o każdym węźle jako oddzielnej pozycji w tablicach routingu oraz propagacja informacji o zmianie punktu podłączenia z wykorzystaniem tradycyjnych protokołów routingu wprowadza znaczne opóźnienia czasowe przy przekazywaniu pakietów oraz zmniejsza efektywność wykorzystania zasobów sieci [13].

Jedynie dwa rozwiązania proponujące dynamiczną zmianę routingu przewidziane zostały do zastosowania w sieciach rozległych z gwarancją realizacji makromobilności (MSM-IP [22] oraz Connexion [23]). Są to jednak rozwiązania specyficzne i ze względu na przyjęte w trakcie projektowania założenia (mimo, że klasyfikuje się je jako rozwiązania makromobilności) mogą efektywnie funkcjonować jedynie w przypadku topologii o ograniczonej złożoności.

Pierwsze z nich zakłada implementację transmisji grupowej (multicast) w całej sieci. Każdy węzeł mobilny posiada przypisany identyfikator grupy multicastowej, a zmiana punktu podłączenia powoduje rozszerzenie grupy multicast o kolejne routery. W związku z problemami w realizacji transmisji grupowej w sieci globalnej, rozwiązanie to ogranicza się w praktyce do pojedynczej domeny zarządzania.

Drugie wspomniane rozwiązanie, Connexion, zaprojektowane zostało przez ekspertów firmy Boeing i jest dedykowane dla przemieszczających się samolotów. Każdy samolot posiada przypisany prefiks sieci, który w przypadku zmiany punktu podłączenia rozgłaszany jest z innego miejsca w topologii sieci.

Oba rozwiązania zapewniające mobilność w oparciu o modyfikację routingu dedykowane są dla rozwiązań mikromobilności. Jakkolwiek domeny mogą być w tym przypadku stosunkowo rozbudowane, jednak charakter wykorzystywanych mechanizmów, a w szczególności niespełnianie odpowiednich wymagań w sieci globalnej, powodują że makromobilność nie może być realizowana poprzez modyfikację routingu w przedstawionych ujęciach.

Jak zatem widać, dysponujemy obecnie (opisanymi już w części pierwszej rozdziału) sprawdzonymi rozwiązaniami makromobilności IP opartymi na odwzorowaniu adresów. W tej sytuacji pojawia się potrzeba zdefiniowana ograniczonego zestawu mechanizmów pozwalających na efektywne zarządzanie urządzeniami mobilnymi przemieszczającymi się na ograniczonym obszarze pojedynczej domeny - co prowadzi do propozycji protokołów mikromobilności. Większość z nich stanowi rozszerzenie funkcjonalności MIP/PMIP, część wykorzystuje doświadczenia telefonii komórkowej. Bogaty przegląd tego typu rozwiązań można znaleźć w [9], [13], [24].

## 6 Metody optymalizacji przełączania protokołów mobilności IP

Konieczność informowania macierzystego agenta (HA) o zmianie lokalizacji stacji ruchomej (MH), obecna w przypadku protokołów Mobile IP, wymusza generację znacznego ruchu sterującego, związanego z przesyłaniem przez sieć pakietów sygnalizacyjnych realizujących procedury rejestracji. Przyczynia się to do znacznego zużycia zasobów sieciowych. Wprowadza też opóźnienia i przerwy w realizacji sesji TCP, mogące skutkować zerwaniem połączenia przez stację CN. Tym samym, w dużej sieci, z dużą liczbą terminali mobilnych przemieszczających się między domenami ze znaczna częstością pojawiają się poważne problemy ze skalowalnością rozwiązania mobilności IP. Jako częściowe rozwiązanie tego problemu zaproponowane zostały protokoły optymalizacji przełączania, z których najpopularniejsze zaprezentowane zostaną w niniejszym rozdziale.

### 6.1 Cellular IP

Propozycja tego protokołu powstała na Uniwersytecie Columbia oraz w laboratoriach firmy Ericsson. Przy tworzeniu rozwiązania Cellular IP [25] autorzy

wykorzystali doświadczenia nabyte podczas budowy sieci telefonii komórkowej. Rozwiązanie to nie jest w zasadzie nowym protokołem i nie zastępuje protokołu MIP, ale jest jego rozszerzeniem. Podnosi ono efektywność obsługi stacji mobilnych w ramach pojedynczej domeny, nazywanej w tym przypadku klastrem. Składa się on z zespołu routerów i punktów dostępowych nadzorowanych przez wspólnego agenta ulokowanego w routerze brzegowym. W przypadku przemieszczenia MH poza dany klaster, używane są klasyczne rozwiązania makromobilności stosowane w opisanym wcześniej protokole MIP.

Podstawowe zadanie Cellular IP to umożliwienie szybkiego przekazywania obsługi (*ang. Fast Handoff*), podczas przemieszczania stacji ruchomych, przy zachowaniu prostoty oraz minimalnej liczby wymienianych pakietów kontrolnych. Najważniejszą cechą, która wyróżnia ten protokół spośród innych, jest położenie dużego nacisku na problem oszczędzania energii przez stację. Dlatego w protokole uwzględniono dwa stany pracy stacji klienckiej:

- aktywny - gdy stacja nadaje i odbiera pakiety,
- oszczędzania energii (pasywny) - gdy stacja nie realizuje transmisji.

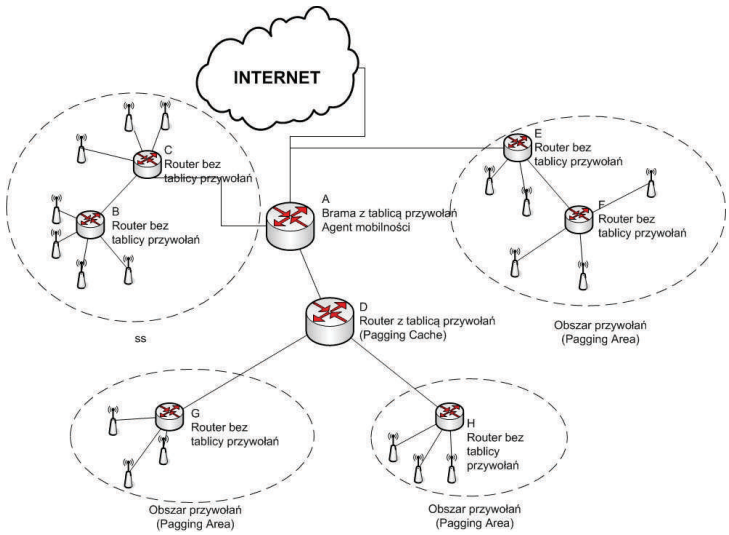
Przejście ze stanu pasywnego do aktywnego może być wywołane potrzebą nadania pakietu przez stację lub koniecznością odbioru pakietu do niej kierowanego.

Sieć typu Cellular IP składa się ze zbioru punktów dostępowych (AP), które udostępniają bezprzewodowy interfejs dostępu do sieci dla ruchomych stacji MH na pewnym obszarze geograficznym. Wysyłają one okresowo sygnały identyfikujące (*ang. Beacon Signal*), na podstawie których stacje MH rozpoznają najbliższy AP.

Punkty dostępu połączone są routerami, z których co najmniej jeden odpowiada za przekazywanie ruchu poza klaster i nazywany jest bramą zewnętrzną - *ang. gateway*. Dla uproszczenia przyjmiemy, że w ramach sieci występuje tylko jeden gateway. Protokół dopuszcza możliwość istnienia wielu takich bram, jednak w takiej sytuacji należy zadbać, aby dana stacja zawsze korzystała z tego samego gateway'a. Bramy pełnią także, z punktu widzenia współpracy z protokołem MIP, rolę agentów mobilności (domowego i obcego).

Twórcy protokołu zauważyli, że podobnie jak w sieciach telefonii komórkowej, nie ma potrzeby, aby ruchome stacje aktualizowały informacje o swoim położeniu przy każdym przemieszczeniu. Analogicznie nie ma potrzeby przechowywania dokładnej informacji o położeniu wszystkich stacji. Dokładną lokalizację należy znać jedynie dla stacji znajdującej się w stanie aktywnym, zaś dla pozostałych stacji tylko "zgrubną" informację o lokalizacji. Jest to możliwe dzięki pogrupowaniu punktów dostępu danej domeny na obszary przywołań (*ang. PA - Paging Area*). Liczba AP składających się na obszar przywołania uzależniona jest od budowy sieci - w wariancie maksymalnej liczby PA każdy taki obszar składa się z jednego AP. Prostszy przypadek uwzględnia sytuację, kiedy w ramach domeny występuje tylko jeden obszar PA. Przykładowa topologia sieci została przedstawiona na Rys. 7.

Lokalizacja stacji odbywa się więc w Cellular IP na podstawie dwóch podsystemów utrzymywania informacji o jej położeniu. Pierwszym z nich są wpisy




**Rysunek 7.** Przykładowa topologia sieci (klastra) Cellular IP.

w tablicach tras routerów pośredniczących, zlokalizowanych między bramą, a AP. Drugim - wpisy w tablicach routerów obszarów przywołań, które są utrzymywane zarówno dla stacji przebywających w stanie aktywnym, jak i oszczędzania energii.

Wpisy dla danej stacji w tablicach tras poszczególnych routerów tworzą ścieżkę. Ścieżki te wymagają okresowego odświeżania, a zatem, kiedy stacja przebywa w stanie aktywnym musi informować sieć o każdym swoim przemieszczeniu (zmianie AP) oraz odświeżać ścieżki, gdy się nie przemieszcza. Dzięki temu, w sposób ciągły, utrzymana zostaje informacja o aktualnej lokalizacji stacji ruchomej.

Gdy MH chce przejść w stan pasywny po prostu przestaje nadawać pakiety. Po upływie czasu ważności trasy (*ang. routing timeout*) jej ścieżka zostaje usunięta - anulowane są wpisy we wszystkich routerach tworzących ścieżkę. Ponieważ czas ważności ścieżek w tablicach routerów obszarów jest większy (*ang. paging timeout*), wpisy te w zostają zachowane. Stacja będzie aktualizować informację o obszarze przywołań, gdy przemieści się do nowego PA lub odświeży wpis po upłynięciu jego czasu aktualizacji (*ang. paging update time*).

Pakiet kierowany z sieci zewnętrznej do stacji ruchomej trafia najpierw do bramy, która szuka adresu domowego stacji MH w tablicy tras. Jeżeli go znajdzie, kieruje pakiet wzdłuż zestawionej do niej ścieżki. W przeciwnym przypadku przeszukuje tablicę obszarów przywołań i po znalezieniu adresu kieruje pakiet do właściwego obszaru PA, w którym znajduje się stacja. Jeżeli nie ma wpisu dla danej stacji, możliwe są dwa scenariusze - jeżeli pakiet dotarł do bramy poprzez

interfejs domyślnego routingu, pakiet jest usuwany; w przeciwnym przypadku jest on przesyłany na interfejs domyślny.

Zanim jednak brama przeszuka tablicę tras sprawdza, czy pakiet nie jest adresowany do niej. Jeśli tak, sprawdza czy nie jest to pakiet tunelowany do jednej ze stacji, dla których jest agentem obcym (stosowany jest tu adres obsługi agenta obcego - *ang. foreign agent care-of address*). Z takiego pakietu zdejmowane są nagłówki tunelowania, a następnie routing odbywa się jak dla pozostałych pakietów.

Pakiety przesyłane przez ruchomą stację przesyłane są według tych samych zasad - router, który odbierze pakiet sprawdza, czy ma wpisy dla danej stacji. Jeżeli nie, wówczas kieruje pakiet w stronę bramy. Jeżeli znajdzie wpis dla danej stacji przekierowuje pakiet na ścieżkę do stacji lub do obszaru PA, w którym przebywa stacja korespondująca.

### 6.2 Handoff-Aware Wireless Access Internet Infrastructure (HAWAII)

Inną propozycją wspierania mikromobilności (opracowaną w Bell Labs) jest HAWAII (*ang. Handoff-Aware Wireless Access Internet Infrastructure*) [26]. Jest to rozwiązanie, które uzupełnia niedostatki MIP, przy czym podobnie do Cellular IP - optymalizuje mikromobilność w sposób przezroczysty dla ruchomych stacji działających z protokołem MIP.

W przypadku HAWAII podstawową jednostką sieci jest domena, odpowiadająca w zasadzie pojęciu klastra i będąca podzbiorem sieci bezprzewodowych, składających się z pewnej liczby punktów dostępowych (AP), które w porównaniu z klastrem Cellular IP poszerzono o funkcję wyznaczania tras. Domena posiada też co najmniej jeden router, wśród których jeden jest routerem głównym domeny (*ang. Domain Root Router*). Pełni on dwie role: bramy pośredniczącej pomiędzy domeną a Internetem oraz agenta domowego (HA) protokołu MIP. Zatem w architekturze HAWAII, zamiast sieci domowej i sieci obcej, występuje pojęcie domeny domowej i domeny obcej, z których każda może obejmować wiele podsieci IP.

Charakterystyczną cechą architektury HAWAII jest brak zmiany adresu obsługi (CoA) w scenariuszu mikromobilności - poruszania się terminala mobilnego (MH) w obrębie domeny. Funkcjonalność taką osiągnięto poprzez podwójny poziom rejestracji MH: pomiędzy ruchomą stacją a AP (z wykorzystaniem klasycznych wiadomości rejestracji MIP) oraz pomiędzy AP a agentem domowym (HA). Dzięki temu z punktu widzenia MH obsługa jest identyczna jak w protokole MIP, zaś zastosowanie zmodyfikowanych procedur pomiędzy AP i resztą systemu pozwala na zmniejszenie liczby pakietów rejestracji wysyłanych do agenta domowego. Aktualizacja informacji o położeniu MH jest konieczna tylko wówczas, gdy stacja zmieni domenę lub gdy upłynie czas rejestracji.

Aby zapewnić poprawne działanie sieci podczas przemieszczenia stacji wprowadzono mechanizm tzw. dynamicznych ścieżek. Służy on do określania położenia stacji w ramach domeny tak, by możliwe było utrzymanie ciągłej komunika-

cji. Gdy ruchoma stacja zmieni punkt dostępu wysyła żądanie rejestracji MIP, które traktowane jest odmiennie z zależności od rodzaju przemieszczenia:

- w przypadku przemieszczenia wewnątrz domeny powoduje jedynie aktualizację ścieżki dynamicznej,
- w przypadku przemieszczenia do nowej domeny HAWAII powoduje otrzymanie nowego adresu obsługi, zestawienie nowej ścieżki dynamicznej i poinformowanie agenta domowego,
- w przypadku przemieszczenia poza domenę HAWAII powoduje otrzymanie nowego adresu obsługi i poinformowanie agenta domowego.

Ścieżki dynamiczne utrzymywane są przez routery w obrębie domeny i są typu *soft-state*, czyli wymagają odświeżania. Czas odświeżania jest zazwyczaj krótszy od czasu rejestracji stacji, dlatego punkt dostępu jest odpowiedzialna za połączenie i okresowo odświeża ścieżki (wpisów w poszczególnych routerach).

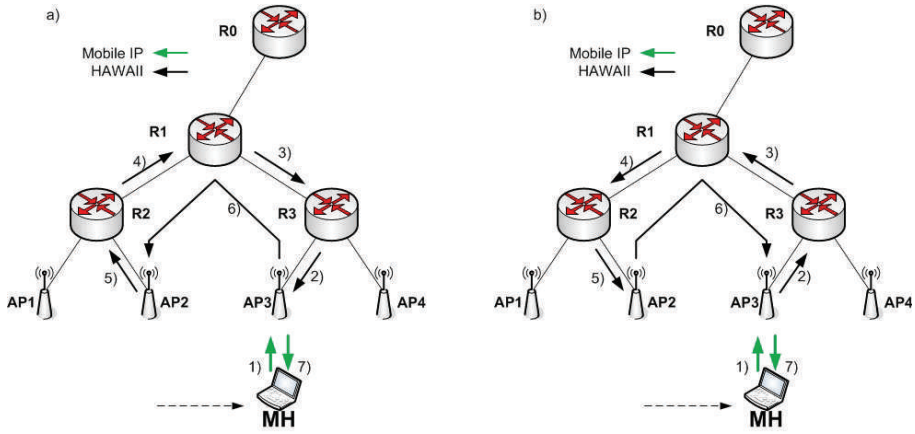
Gdy ruchoma stacja przemieszcza się w ramach domeny (mikromobilność), zwraca się o obsługę do nowego punktu obsługi (agenta obcego, praktycznie - do AP) i przesyła mu żądanie rejestracji. Następnie AP przeprowadza procedurę aktualizacji ścieżki do stacji stosując jeden z dwóch schematów: z przekierowaniem pakietów przez poprzedniego agenta (*ang. Forwarding Path Setup Scheme*), bądź bez przekierowania pakietów przez poprzedniego agenta (*ang. Non-Forwarding Path Setup Scheme*).

W przypadku przekierowywania pakietów przez poprzedniego agenta pakiety są przesyłane od poprzedniego do nowego AP za pośrednictwem tzw. routera "przecięcia" (ang. cross-over router), czyli ostatnie wspólnego routera dla tras: pomiędzy węzłem korespondującym i poprzednim AP oraz pomiędzy węzłem korespondującym, a nowym AP. W rezultacie, gdyby poprzedni AP odebrał pakiety adresowane do MH w trakcie rekonfiguracji ścieżki, przekierowuje je już do nowego AP w celu dostarczenia do MH.

Wybór jednego ze schematów związany jest bezpośrednio z konfiguracją sieci. Pierwszy z nich przeznaczony jest dla sieci, w których ruchoma stacja potrafi współpracować tylko z jednym punktem obsługi w tym samym czasie (np. stosujących dostęp z podziałem czasowym TDMA), zaś drugi jest wskazany dla sieci, w których ruchoma stacja może równocześnie współpracować z wieloma AP (np. stosujących dostęp z podziałem kodowym CDMA), czyli realizować multihoming.

Na rys. 8 zilustrowano działanie protokołu HAWAII dla obu algorytmów przekierowywania pakietów - forwarding (z przekierowaniem) oraz non forwarding (bez przekierowania). W scenariuszu z przekierowywaniem (Rys. 8a) AP3 po odebraniu pakietu rejestracji od MH (1) odczytuje z niego adres poprzedniego punktu dostępowego na podstawie zawartego rozszerzenia powiadomienia poprzedniego agenta. Następnie wysyła pakiet aktualizacji ścieżki (HANDOFF UPDATE MESSAGE - 2) bezpośrednio do poprzedniego punktu obsługi AP2. Ten odnotowuje, że pakiety adresowane do MH należy kierować przez R3 i przesyła pakiet HANDOFF UPDATE MESSAGE do routera R2 (3), a ten do następnego routera R1 (4), który jest router przecięcia. Router przecięcia przesyła dalej pakiet do AP3 (5), przy czym wszystkie routery po drodze aktualizują

swoje wpisy dla danej stacji w tablicach tras. Dopiero, gdy AP3 odbierze pakiet (6) zmienia swój wpis w tablicy tras i przesyła potwierdzenie (REGISTRATION REPLY - 7) do MH.



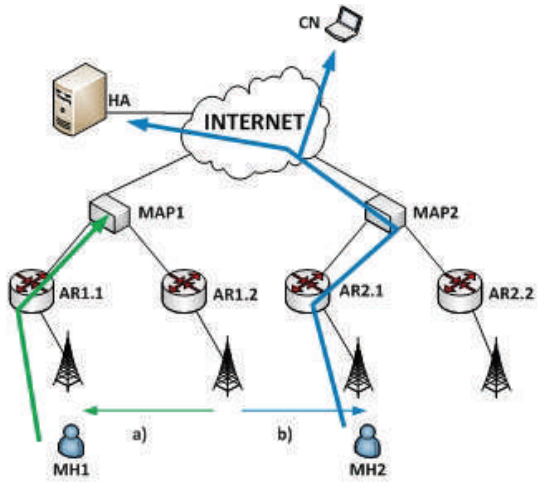

**Rysunek 8.** Działanie protokołu HAWAII podczas przemieszczenia w ramach domeny a) schemat z przekierowaniem pakietów przez poprzedniego agenta, b) schemat bez przekierowania pakietów przez poprzedniego agenta.

Niezależnie od zastosowanego scenariusza ścieżka zostanie zaktualizowana i pakiety adresowane do stacji po dotarciu od jednego z routerów znajdujących się na jej ścieżce zostaną przekierowane do jej bieżącej lokalizacji.

### 6.3 Hierarchical Mobile IPv6 (HMIPv6)

Propozycja Hierarchical Mobile IPv6 (HMIPv6) [27] jest rozszerzeniem protokołu MIP pozwalającym na zwiększenie efektywności strategii mikromobilności poprzez wprowadzenie dodatkowych elementów do infrastruktury znajdujących się bliżej użytkownika mobilnego (Rys. 9). Elementy te nazywane są Mobility Anchor Point (MAP) i posiadają funkcjonalność podobną do funkcjonalności agentów domowych oraz mogą być usytuowane na każdym poziomie hierarchii topologii sieciowej, tworząc podsieci nazywane regionami. W przypadku przełączania między sieciami dostępowymi podlegającymi temu samemu MAP sygnalizacja realizowana jest jedynie pomiędzy MH i MAP, a nie pomiędzy MH i agentem domowym. Zyskiwana jest w efekcie redukcja czasu rejestracji oraz wielkości ruchu sygnalizacyjnego, a w efekcie poprawa skalowalności całego systemu.

W ramach opisywanego rozwiązania pojedynczy adres obsługi (Care of Address - CoA) zastąpiony został ich parą - Regional Care-of-Address (RCoA), który identyfikuje daną podsieć i przypisany jest do elementu MAP oraz On-link




**Rysunek 9.** Scenariusz wykorzystania protokołu Hierarchical Mobile IPv6; 1) przełączenie w obrębie jednego regionu; 2) przełączenie między regionami.

Care of- Address (LCoA), który obowiązuje jedynie w podsieci danego MAP. Jeśli terminal mobilny zmieni punkt podłączenia w obrębie podsieci przyporządkowanych do jednego MAP, zmieni jednocześnie router dostępowy (Access Router - AR) oraz adres LCoA. Natomiast identyfikator regionu (RCoA), w którym jest on aktualnie zarejestrowany nie zostanie zmieniony. Dzięki czemu mobilność terminala ruchomego w obrębie regionu nie jest widoczna dla węzłów korespondujących oraz agenta domowego HA protokołu MIP - nie jest wymagane wysyłanie do nich widomości Binding Update.

Na rysunku 9 zostały zaprezentowane scenariusze przełączenia w obrębie jednego regionu (a) oraz pomiędzy nimi (b). Na rysunku zaprezentowano strukturę sieci z jednym poziomem elementów MAP, lecz rozwiązanie HMIP pozwala na budowę hierarchicznych struktur wielopoziomowych.

Opisywane rozwiązanie wymaga umieszczenia elementów systemu po stronie terminala mobilnego. Jeśli terminal ruchomy nie posiada takiego rozszerzenia, zostanie zastosowany standardowy protokół MIPv6.

### 6.4 Fast Handovers for Mobile IPv6 (FMIPv6)

Protokół Mobile IPv6 Fast Handovers, przedstawiony w RFC 5268 [28], a uzupełniony o obsługę IPv4 w RFC 4988 [29], umożliwia dostarczanie informacji konfiguracyjnej dotyczącej nowej sieci do terminala mobilnego jeszcze przed jego odłączeniem od sieci aktualnie wykorzystywanej. Podczas przygotowania do przełączenia wykorzystywane są wiadomości *Router Solicitation for Proxy Advertisement* (RtSolPr) oraz *Proxy Router Advertisement* (PrRtAdv). Pierwsza z nich jest żądaniem przesyłanym przez terminal mobilny do obecnego ro-

utera dostępowego, zwierającym zapytanie o parametry routera dostępowego po przełączeniu. Druga z nich zawiera odpowiedź na to zapytanie i może być przyczynkiem dla samego przełączenia. Dzięki tym dwóm wiadomościom terminal mobilny może uzyskać nowy adres obsługi IPv6 obowiązujący w nowej sieci (CoA) jeszcze podczas połączenia do aktualnego punktu dostępowego. W ten sposób opóźnienie przełączenia zostało zredukowane o czas wymagany na uzyskanie adresu IP. Router dostępowy nowej sieci może również zawczasu przeprowadzić procedurę wykrycia duplikacji adresów (*Duplicate Address Detection* - DAD), wyręczając w tym terminal mobilny, co również korzystanie wpływa na ograniczenie czasu przełączania.

W protokole FMIPv6 wyszczególnione zostały dwa tryby pracy. Pierwszy z nich to tryb predykcyjny (*predictive*), w którym procedura rejestracji (zgodna z protokołem MIPv6) w nowej sieci wykonywana jest jeszcze w trakcie podłączenia do obecnej. Drugi tryb, reaktywny (*reactive*), przewiduje, że rejestracja ta wykonywana jest dopiero po zakończeniu przełączenia do nowej sieci.

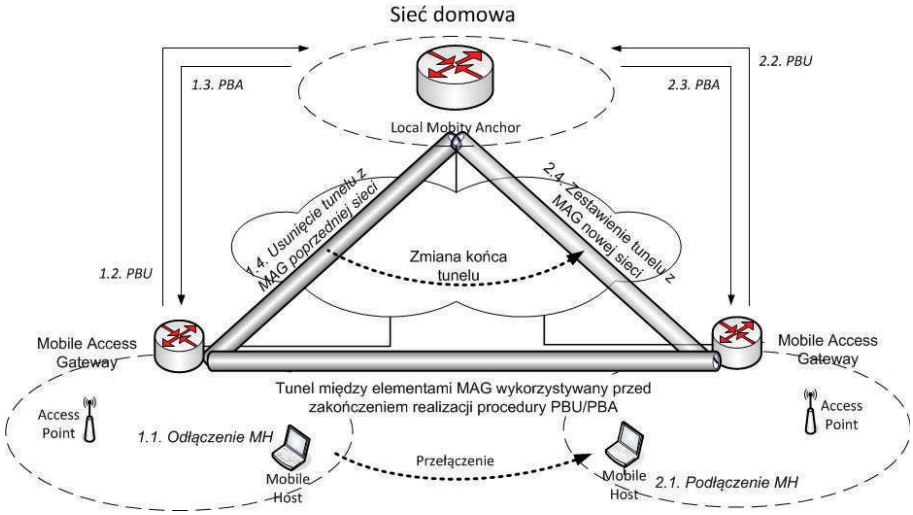
### 6.5 Fast Handovers for Proxy Mobile IPv6

Jedną z optymalizacji protokołu Proxy MIPv6 jest zaproponowany w RFC 5949 protokół Fast Handovers for Proxy Mobile IPv6 (FPMIPv6) [30], który jest przełożeniem idei opisanego powyżej rozwiązania FMIPv6 do scenariusza mobilności zapewnianej przez elementy infrastruktury sieci. Ponieważ rozwiązania PMIPv6 [21] nie są realizowane przy współpracy z terminalem klienta, również ich optymalizacje nie mogą zakładać współpracy ze strony urządzenia mobilnego. Wynika stąd, że w tym przypadku wiadomości FMIPv6, takie jak np. PrRtAdv czy RtSolPr nie mogą zostać użyte.

Celem tego rozszerzenia, podobnie jak ma to miejsce w przypadku FMIPv6, jest optymalizacja przełączania polegająca na skróceniu czasu jego trwania oraz redukcji liczby utraconych w tym procesie pakietów. Zaproponowane zostało wykorzystanie tunelu pomiędzy elementami pMAG (poprzedni MAG) oraz nMAG (następny MAG).

Tunel ten wykorzystywany do dostarczania ruchu przeznaczonego dla terminala mobilnego w okresie po jego podłączeniu do nowego punktu dostępowego (podłączenie w warstwie 2 ISO-OSI), a jeszcze przed zakończeniem realizacji procedury wymiany, opisanych w pierwszej części rozdziału, wiadomości Proxy Binding Update (PBU) i Proxy Binding Acknowledgment (PBA) (rys. 10). Zakończenie wymiany wiadomości PBU/PBA spowoduje dostarczanie ruchu bezpośrednio przez nMAG - do tego czasu ruch kierowany jest do urządzenia pMAG, które wykorzystuje wspomniany tunel aby dostarczyć go do urządzenia mobilnego w jego nowej lokalizacji.

Dodatkowo, w urządzeniach pMAG/nMAG może zostać zrealizowane buforowanie pakietów, które niweluje ich straty. Oczywiście nakłada to dodatkowe wymagania zapewnienia odpowiedniej ilości zasobów w elementach MAG. Istnieją dwa warianty opisywanego rozwiązania - predykcyjny (predictive) i reaktywny (*reactive*). W pierwszym z nich o przełączeniu decyduje element pMAG, natomiast w drugim element nMAG.



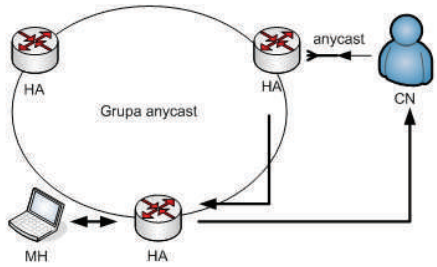

**Rysunek 10.** Schemat architektury protokołu Fast Handover for Proxy Mobile IPv6.

### 6.6 Global Home Agent to Home Agent (GHAHA)

Obok rozwiązań, które usprawniają przełączanie w ramach jednej domeny proponowane są również takie, które zwiększają efektywność przełączania protokołów makromobilności. Przykładem może być protokół Global Home Agent to Home Agent (GHAHA) [31], który początkowo został zaproponowany, jako rozwiązanie pozwalające uniknąć routingu trójkątnego w przypadku rozwiązań opartych na protokole MIP. Ideą tego rozwiązania jest topologiczne rozproszenie elementów HA, z których każdy posługuje się tym samym adresem typu anycast dla potrzeb obsługi rejestracji terminali mobilnych oraz obsługuje adresy domowe z tego samego prefiksu IPv6 - również typu anycast.

Terminal mobilny rejestruje się do grupy HA używając jej wspólnego adresu anycast (Rys. 11), co przekłada się na przeprowadzenie rejestracji u agenta HA znajdującego się najbliżej niego. Po zakończeniu procedury rejestracji, ten konkretny HA staje się jego głównym agentem domowym. Do wszystkich pozostałych agentów domowych wysyła on powiadomienie zawierające powiązanie [M,H], gdzie M oznacza terminal mobilny, a H agenta HA który przeprowadził rejestrację (a zatem jest elementem HA najbliższym terminala mobilnego). Poruszający się terminal może zmienić punkt podłączenia, co spowoduje zmianę jego głównego agenta HA, w miarę jak mechanizmy dostarczania pakietów typu anycast będą kierowały żądania rejestracji MH do różnych (zawsze najbliższych) agentów HA.

Oprócz zakładanych korzyści, wynikających z uproszenia routingu, rozwiązanie GHAHA pozwala również na realizację koncepcji Distributed Mobility Ma-




**Rysunek 11.** Ilustracja wymiany danych dla rozwiązania Global Home Agent to Home Agent.

nagement (DMM). Umożliwia ona zwiększenie skalowalności innych rozwiązań, z którymi propozycja GHAHA zostanie zintegrowana (np., MIP, PMIP) [32].

## 7 Inne koncepcje mechanizmów mobilności służące obsłudze użytkowników ruchomych

Poza opisanymi powyżej rozwiązaniami, bazującymi na mechanizmach warstwy sieciowej, opracowano również szereg mechanizmów alternatywnych, ulokowanych w wyższych warstwach modelu ISO-OSI. Rozwiązania tego typu są z reguły łatwiejsze we wdrożeniu, gdyż nie wymagają modyfikacji bazowych mechanizmów kontrolujących funkcjonowanie sieci. Wadą jest natomiast ograniczone spektrum ich działania, zarówno jeśli chodzi o zasięg rozwiązania (skalowalność), jak i np. obsługa wyłącznie wybranej grupy aplikacji.

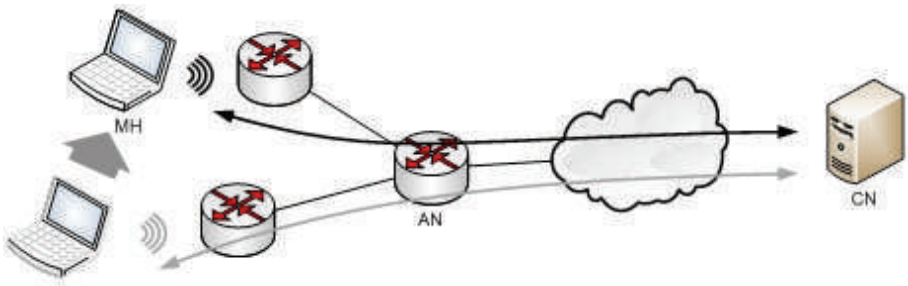
### 7.1 Translacja adresów

Na uwagę zasługują dwa rozwiązania mobilności, które korzystają z translacji adresów NAT - wcześniejsze RAT [33] i opracowane później: MobileNAT [34]. W obu przypadkach mechanizm translacji adresów został wykorzystany w celu zastąpienia tunelowania. MobileNAT pozwala uzyskać w ten sposób zarówno mikro, jak i makromobilność, podczas gdy RAT, realizujący mobilność wyłącznie w skali mikro, jest dostosowany do obsługi węzłów bez wsparcia dla mobilności, o niezmodyfikowanym stosie sieciowym.

Rozwiązanie MobileNAT oparte jest na wykorzystaniu dwóch adresów IP - jednego nazywanego wirtualnym (Av), oraz drugiego - dowolnego adresu IP uzyskanego lub skonfigurowanego w sieci dostępowej, nazywanego dalej adresem fizycznym (Ap). Cechą adresu wirtualnego jest jego niezmienność, natomiast fizyczny adres IP zmienia się zgodnie z miejscem podłączenia terminala mobilnego. Ponieważ adresy te z definicji należą do różnych podsieci IP, wprowadzane jest pojęcie Anchor Node (AN) - węzła odpowiedzialnego za styk tych dwóch adresacji z punktu widzenia dostępu węzła mobilnego do sieci Internet, i tym

samym obsługę scenariuszy mobilności w skali makro. Rozwiązanie skupia się wokół utrzymania trwających sesji nawiązanych z punktu widzenia terminala mobilnego (MH) z węzłem korespondującym (CN). Stos sieciowy węzła mobilnego został zmodyfikowany w taki sposób, że aplikacje pracują z wykorzystaniem adresu wirtualnego Av, który przed jego opuszczeniem podlega translacji na aktualny adres fizyczny Ap. W ten sposób stos sieciowy terminala mobilnego jest uniewrażliwiony na zmiany fizycznego adresu IP wynikającego z sieci dostępowej, w której aktualnie przebywa MH.

W przypadku scenariusza mikromobilności, kiedy MN przemieszcza się w obrębie sieci podległej pojedynczemu Anchor Node (AN), kwestia zmiany adresu, na który wysyłane są datagramy IP do MH jest rozwiązywana w AN. Po przełączeniu aktualizowana jest tablica translacji w AN ze starego adresu fizycznego Ap1 do nowego Ap2 terminala mobilnego. Zmiana tego odwzorowania powoduje, że z punktu widzenia węzła korespondującego CN adres MH nie ulega zmianie. Sytuację taką ilustruje rysunek 12.

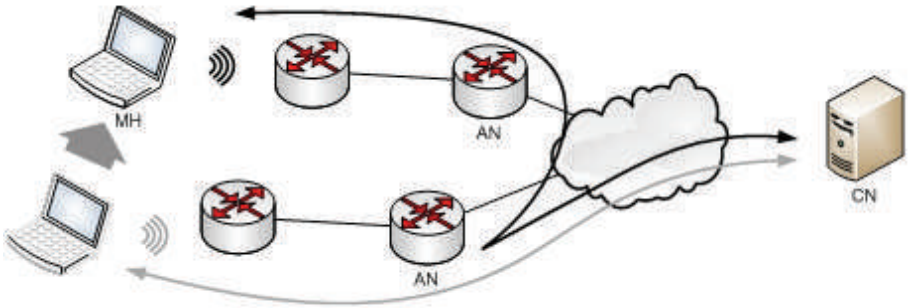



**Rysunek 12.** Mikromobilność w rozwiązaniu MobileNAT.

Nieco inaczej realizowana jest makromobilność, gdy MH przemieszcza się pomiędzy domenami kontrolowanymi przez różne AN. Wówczas pierwotny AN jest odpowiedzialny za translację adresów docelowych dla pakietów pochodzących od CN oraz źródłowych dla pakietów pochodzących od MH. Sytuacja ta najlepiej obrazuje zastąpienie w tym przypadku tunelowania mechanizmem translacji adresów, co jest charakterystyczne dla rozwiązania MobileNAT. Ilustracja scenariusza makromobilności przedstawiona jest na Rys. 13.

Nieco prostsze podejście zaprezentowano w przypadku rozwiązania RAT [33]. Motywacją była ciągle relatywnie niska, popularność protokołów realizacji wsparcia mobilności w warstwie sieciowej, a tym samym mobilności niezależnej od warstw niższych modelu OSI. Autorzy zdecydowali się na opracowanie rozwiązania o ograniczonych możliwościach, lecz łatwego w implementacji, celem spopularyzowania obsługi mobilności w systemach IP.

Rozwiązanie RAT oparte jest o unikalną koncepcję rejestracji terminala mobilnego w sieciach domowej i odwiedzanej. Rejestracja dokonywana jest ręcznie




**Rysunek 13.** Makromobilność w rozwiązaniu MobileNAT.

przez użytkownika, w warstwie aplikacji, za pomocą np. przeglądarki WWW. Jest to z jednej strony uproszczenie strony terminala mobilnego - nie jest tutaj wymagana żadna ingerencja w stos sieciowy - z drugiej natomiast stanowi istotne ograniczenie w procesie przełączania. RAT nie zapewnia zatem obsługi przełączeń w czasie rzeczywistym. Stanowi więc rozwiązanie nomadyczności, co jest ceną za małe skomplikowanie zastosowanych mechanizmów.

Z drugiej strony, RAT porusza kwestie skalowalności, posługując się w samych sieciach dwoma dodatkowymi urządzeniami pełniącymi rolę agentów. Pierwsze z nich to serwer rejestracji (RS), odpowiedzialny za obsługę żądań użytkowników zarejestrowania się w sieci. Drugim komponentem RAT jest urządzenie RAT (*RAT device* - *RD*), które pełni funkcję agenta domowego. Funkcja RD jest w tym przypadku o tyle nietypowa że, podobnie jak w przypadku MobileNAT, zamiast tunelować ruch do terminala mobilnego znajdującego się poza siecią domową wykonywana jest translacja adresów IP. Autorzy sugerują też, że w ten sposób bardzo popularne urządzenia tak czy inaczej pełniące rolę translatorów NAT niewielkim kosztem mogą pełnić funkcje realizujące nomadyczność węzłów. Koncepcyjnie sam mechanizm translacji przedstawia się bardzo podobnie do rozwiązania MobileNAT.

Warto też zauważyć, iż opracowano łatwy sposób migracji z nieskomplikowanego rozwiązania RAT do w pełni funkcjonalnego protokołu MobileIP z obsługą przełączeń w czasie rzeczywistym. Tym samym istnieje płynną ścieżkę przejścia od braku mobilności w warstwie IP, poprzez łatwe we wdrożeniu rozwiązanie częściowe, aż do dojrzałych koncepcyjnie rozwiązań oferujących zaawansowaną funkcjonalność.

Kolejnym z interesujących rozwiązań w grupie wykorzystującej translację adresów jest, opatentowane w Stanach Zjednoczonych oraz pół roku później w krajach Europy, rozwiązania mobilności w warstwie sieciowej nie wykorzystującego tunelowania (enkapsulacji). Patent zatytułowany *Non encapsulation mobile IP* został zgłoszony w roku 2005 [35].

### 7.2 Rozwiązania warstwy transportowej

Przykładem rozwiązania obsługi mobilności wykorzystującego mechanizmy warstwy transportowej jest End-to-End Mobility (E2E) [36]. Podstawą zaproponowanego rozwiązania jest stworzenie rozszerzenia protokołu TCP pozwalającego na utrzymanie ciągłości wymiany danych nawet po zmianie adresu IP związanego ze zmianą punktu podłączenia terminala mobilnego do sieci. Jako identyfikator terminala mobilnego wykorzystywana jest jego nazwa domenowa DNS. Po zmianie punktu podłączenia, korzystając z procedur Dynamic DNS, terminal mobilny dokonuje aktualizacji rekordu typu A (IPv4) lub AAAA (IPv6) odzwierciedlającej zmianę jego aktualnego adresu IP. Nawiązanie nowych połączeń odbywa się, od tego momentu, z użyciem zaktualizowanego adresu IP. Utrzymanie trwających już połączeń TCP wymaga zastosowania dodatkowych mechanizmów.

Połączenie TCP identyfikowane jest przez cztery parametry: źródłowy adres IP, port źródłowy, docelowy adres IP oraz port docelowy. Zmiana, w trakcie trwania połączenia TCP, któregokolwiek z powyższych parametrów uczestniczących w nim węzłów spowoduje jego przerwanie. W celu zachowania ciągłości połączenia (rys. 14) proponowane jest wykorzystanie dodatkowej wiadomości Migrate SYN, zaliczanej do aktualnego połączenia i pozwalającej na modyfikację jego parametrów. Dodatkowo w celu zapewnienia bezpieczeństwa zaproponowane zostało wykorzystanie kryptograficznie generowanych tokenów, będących, obok wspomnianych czterech parametrów, identyfikatorami połączenia. Token pozwala jednoznacznie stwierdzić, czy strona próbująca dokonać zmiany parametrów jest rzeczywiście stroną w nim uczestniczącą.

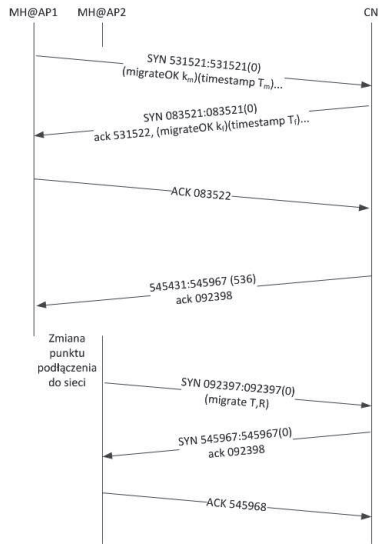
Do popranego działania systemu rozwiązanie E2E musi być zaimplementowane w stosie protokołów TCP/IP wszystkich komunikujących się węzłów.

### 7.3 Rozwiązania aplikacyjne

Innym podejściem do realizacji usługi mobilności jest zastosowanie dedykowanych rozwiązań w warstwie aplikacyjnej. Rozwiązania tego typu cieszą się dość dużą popularnością, ze względu na stosunkowo niewielką dostępność, w praktycznie spotykanych systemach sieciowych, rozwiązań obsługi mobilności pracujących w niższych warstwach modelu ISO-OSI. Ich głównym ograniczeniem jest fakt, iż każda aplikacja musi sama zadbać o implementację mechanizmów mobilności na swoje potrzeby. Poza narzutem implementacyjnym wiąże się to również z niebezpieczeństwem powstania ogromnej liczby niestandaryzowanych i niezgodnych ze sobą rozwiązań, różniących się nawet pomiędzy różnymi implementacjami tej samej usługi. Dąży się zatem, w miarę możliwości, do opracowania możliwie uniwersalnych rozwiązań.

Przykładem takiego podejścia może być protokół Session Initiation Protocol (SIP) [37]. Został on zaprojektowany do nawiązywania oraz zarządzania sesją, w trakcie której następuje wymiana informacji pomiędzy aplikacjami. Najczęściej jest to sesja wymiany danych multimedialnych.

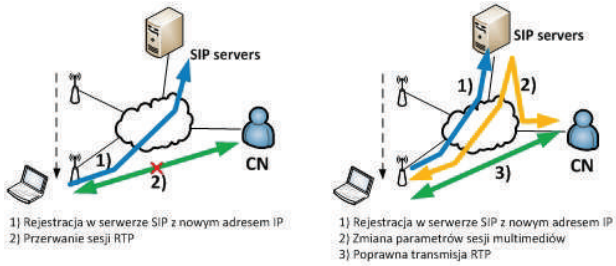
Dzięki swojej uniwersalności, protokół SIP zyskał dużą popularność. Jest często wykorzystywany jako protokół sygnalizacyjny w systemach VoIP, a zbiór




**Rysunek 14.** Zmiana parametrów połączenia TCP z wykorzystaniem mechanizmów E2E.

dotyczących go dokumentów standaryzacyjnych IETF obejmuje ponad 140 pozycji.

Protokół SIP zapewnia mechanizmy umożliwiające komunikację zarówno w przypadku nomadyczności węzłów (gdy ich adresy IP mogą ulegać zmianie, lecz nie zależy nam na zachowaniu ciągłości trwających już sesji komunikacyjnych) jak i pełnej ich mobilności (gdy trwające w momencie przełączenia sesje muszą zostać zachowane). Porównanie dwóch powyższych scenariuszy zostało zaprezentowane na Rys. 15.




**Rysunek 15.** Zapewnienie obsługi nomadyczności i mobilności w warstwie aplikacji - protokół SIP.

Pierwszy scenariusz realizowany jest poprzez procedury rejestracji i wyszukiwania użytkownika. W rozwiązaniach SIP użytkownik identyfikowany jest przez adres Uniform Resource Identifier (URI) w postaci *sip:nazwa@domena*. W procesie rejestracji identyfikator ten zostaje powiązany z aktualnym adresem IP terminala. Wywołanie tego użytkownika skierowane jest do serwerów SIP operatora, który na podstawie powiązania zna adres IP terminala i może skierować do niego wiadomość SIP. Większy problem stanowi zapewnienie pełnej mobilności w trakcie trwania sesji. W tym przypadku konieczna jest modyfikacja parametrów trwającej sesji multimedialnej [38], realizowana z wykorzystaniem dedykowanych do wiadomości SIP np. re-INVITE lub REFER. Należy zauważyć, że zapewnienie mobilności w trakcie trwania połączenia wymaga narzutów sygnalizacyjnych realizowanych dla każdego połączenia oddzielnie. Dodatkowo każda z aplikacji posiadających aktualnie aktywne połączenia musi niezależnie dbać o zapewnienie jego ciągłości w trakcie przemieszczania. Powoduje to oczywiście ograniczoną skalowalność tego rozwiązania.

## 8 Propozycje sieci Internetu Przyszłości ukierunkowane na obsługę mobilności

Wykorzystywane w obecnym Internecie protokoły stosu TCP/IP, zaprojektowane przy założeniu połączeniowego charakteru przekazu pakietów między stacjonarnymi, bądź wolno zmieniającymi położenie terminalami, nie są w stanie sprostać oczekiwaniom Internetu Przyszłości. Jak pokazano w przeprowadzonej analizie, ich wykorzystanie nie rozwiązuje satysfakcjonująco wielu problemów stojących na drodze do efektywnej obsługi zróżnicowanych grup użytkowników oraz usług obecnych w aktualnie funkcjonującym i stale rozwijanym Internecie. Podejmowane są zatem próby zaprojektowania nowych mechanizmów sieciowych, które nie będą modyfikacją istniejących protokołów, a od samego początku zostaną zaprojektowane z uwzględnieniem wymagań użytkowników mobilnych.

Dominującym trendem są tu rozwiązania wykorzystujące zróżnicowane odmiany technik wirtualizacyjnych, których charakterystyczną cechą jest wprowadzenie elementu pozwalającego na odseparowanie mechanizmów warstw wyższych od tzw. sieci transportowej. Sieć transportowa może być w praktyce dowolnym systemem komunikacyjnym, umożliwiającym transmisję danych pomiędzy dwoma urządzeniami -stosowane są tu zarówno rozwiązania warstwy łącza danych (jak np. Ethernet), jak i sieciowej (np.: IPv4, IPv6).

Dzięki zastosowaniu rozwiązań wirtualizacyjnych, techniki stosowane w sieci transportowej, a także jej struktura oraz konfiguracja nie mają bezpośredniego wpływu na logiczną strukturę systemu postrzeganą przez warstwy wyższe. Co więcej, wszystkie powyższe aspekty sieci transportowej mogą być zmieniane w sposób praktycznie dla nich niezauważalny. Czyni to z nich rozwiązania wysoce elastyczne we wdrożeniu.

Kolejną z zalet jest możliwość zastosowania dowolnego, abstrakcyjnego identyfikatora obiektu docelowego, który nie jest determinowany jego lokalizacją w fizycznej strukturze sieci. Jest to ogromna zaleta w przypadku obsługi urządzeń

mobilnych, jako że tego rodzaju identyfikator w naturalny sposób może pozostawać niezmienny. Co więcej, identyfikatory można przypisywać nie tylko urządzeniom sieciowym, lecz także innym zasobom jak np. konkretne dane, osoby, czy też całe grupy tego rodzaju obiektów. Pozwala to na łatwe wprowadzanie w tego rodzaju systemach elementów sieci świadomych przekazywanych treści (Content Aware Networks - CANs).

### 8.1 Host Identity Protocol (HIP)

Protokół Host Identity Protocol (HIP) [39] wprowadza dodatkową warstwę pomiędzy warstwę transportową, a warstwę sieciową oraz przypisuje do węzła kryptograficznie generowany klucz publiczny. Proponowane podejście wprowadza rozgraniczenie identyfikatora (klucz publiczny) węzła oraz jego lokalizatora (adres IP). W praktyce zamiast całego klucza publicznego, dla celów adresowania węzłów wykorzystuje się jego skrót nazywany Host Identity Tag (HIT).

Powiązanie pomiędzy identyfikatorem, a lokalizatorem przechowywane jest w dedykowanych elementach infrastruktury nazywanych serwerami Rendezvous (Rendezvous Server - RVS). Każdy z węzłów posiada przypisany sobie serwer RVS, który nadzoruje jego aktualne położenie. Węzeł korespondujący, aby nawiązać połączenie z MH, odpytuje serwer DNS o informację na jego temat. W odpowiedzi otrzymuje adres IP serwera RVS obsługującego dany terminal.

Do wskazanego serwera RVS kierowany jest następnie pierwszy pakiet transmisji, który jest przez niego przekazywany jest do docelowego MH. W wyniku jego przekazania węzeł korespondujący i mobilny mogą realizować komunikację bezpośrednio (zawiera on aktualne adresy IP obu korespondentów).

Po zmianie punktu podłączenia do sieci, terminal mobilny informuje o tym fakcie przypisany do niego serwer RVS. Na Rys. 16 zostało zaprezentowane porównanie uproszczonych stosów sieciowych - tradycyjnego TCP/IP oraz HIP. Wyróżniona została dodatkowa warstwa pomiędzy warstwami sieciową i transportową.

### 8.2 Locator/ID Separation Protocol (LISP)

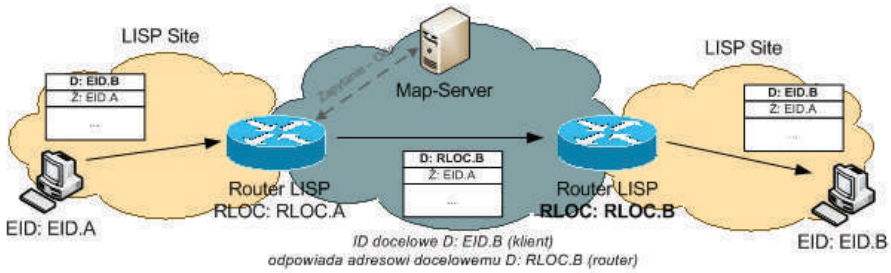
Mechanizm obsługi węzłów mobilnych LISP-Mobility [40] bazuje na wykorzystaniu protokołu Locator/ID Separation Protocol (LISP) [41]. Wspomniany protokół został stworzony w celu poprawy elastyczności i skalowalności routingu IP i wprowadza, w miejsce pojedynczego adresu IP, parę niezależnych parametrów: identyfikator węzła (Endpoint Identifier - EID) oraz informację lokalizacyjną (Route Locator - RLOC). Powyższa koncepcja pozwala na eliminację jednego z najpoważniejszych ograniczeń adresacji IP, a mianowicie powiązania identyfikatora ("tożsamości") węzła z jego aktualną lokalizacją w strukturze sieci IP.

W przypadku klasycznych mechanizmów warstwy sieciowej IP, węzły musiały każdorazowo uzyskać adres IP z puli dostępnej w danej lokalizacji, zdeterminowanej strukturą routingu. W praktyce każda zmiana lokalizacji węzła, skutkująca zmianą jego punktu dostępu do sieci, powodowała zatem konieczność uzyskania nowego adresu IP, a co zatem idzie - zmianę jego tożsamości. Rozwiązanie LISP

**Rysunek 16.** Porównanie stosów sieciowych architektury TCP/IP oraz HIP.

(Rys. 17) umożliwia zachowanie niezmiennego identyfikatora węzła (EID) wykorzystywanego przez warstwy wyższe dla celów komunikacji z nim, podczas gdy warstwa sieciowa posługuje się informacją lokalizacyjną, podlegającą zmianie w miarę zmiany lokalizacji węzła, zawartą w parametrze RLOC.

Istotnym postulatem podczas projektowania protokołu LISP, było zachowanie kompatybilności z istniejącymi rozwiązaniami sieciowymi, stąd zarówno EID jak i RLOC mają postać adresów IP. Ponadto, w swojej bazowej wersji, rozwiązanie nie wymaga modyfikacji stosu sieciowego węzłów końcowych, gdyż wszystkie mechanizmy zlokalizowano na routerach dostępowych.




**Rysunek 17.** Schemat działania protokołu LISP.

W momencie, gdy urządzenie zamierza przesłać dane do innego, tworzy pakiet IP wykorzystując identyfikatory EID w polach adresu źródłowego i docelowego. Router dostępowy, obsługujący dany węzeł źródłowy (tzw. Ingress Tunnel Router - ITR), otrzymawszy tego rodzaju pakiet, komunikuje się z serwerem typu Map-Server. Zadaniem tego urządzenia jest tworzenie i przechowywanie

odwzorowań identyfikator EID - informacja lokalizacyjna RLOC. Dla celów skalowalności, informacja RLOC przechowywana przez Map-Server nie jest unikalna dla każdego z EID, lecz wskazuje na router dostępowy (wspierający mechanizmy LISP) odpowiedzialny za obsługę danej grupy węzłów docelowych (tak zwanego LISP Site).

Dysponując tą informacją ITR wysyła otrzymany pakiet do routera dostępowego obsługującego węzeł przeznaczenia (Egress Tunnel Router - ETR), stosując mechanizmy enkapsulacji. ETR rozpakowuje pakiet i dostarcza go do węzła docelowego.

W efekcie dysponujemy skalowalnym rozwiązaniem pozwalającym na przezroczyste dla warstw wyższych przenoszenie niezmodyfikowanych węzłów pomiędzy lokalizacjami sieciowymi, bez zmiany ich informacji adresowej, co znajduje cały szereg zastosowań, np. poprawia elastyczność wykorzystania dostępnych pul adresowych, pozwala na łatwą zmianę lokalizacji elementów infrastruktury usługowej, umożliwia szybką reakcję w razie awarii itp. Dodatkowo trzeba zauważyć, iż rozwiązanie to jest również wygodnym narzędziem migracji, gdyż warstwa sieciowa (odpowiedzialna za dostarczanie danych) może zostać zmieniona, bez konieczności zmian w mechanizmach warstw wyższych.

Należy jednak pamiętać, iż rozwiązanie LISP nie jest bezpośrednio wymieniane jako metoda obsługi mobilności węzłów, ponieważ obszar jego działania byłby ograniczony wyłącznie do sieci wyposażonych w routery dostępowe rozbudowane o mechanizmy wymagane przez to rozwiązanie. Dla celów obsługi węzłów mobilnych zalecana jest modyfikacja protokołu LISP, a mianowicie LISP-Mobility.

LISP-Mobility rezygnuje po części ze skalowalności oferowanej przez opisane powyżej rozwiązanie, gdyż wymaga aby każdy z węzłów mobilnych stanowił autonomiczny LISP Site. Wówczas każdy z węzłów mobilnych dysponuje utrzymywanym w urządzeniu Map-Server odwzorowaniem EID-RLOC, które na bieżąco aktualizuje.

Tego rodzaju podejście umożliwia realizację makromobilności, lecz z drugiej strony wymaga ulokowania elementów systemu LISP-Mobility w węźle mobilnym MN (nie ma przy tym takiej konieczności w przypadku CN), a także przekłada się na znacznie większą liczbę odwzorowań utrzymywanych przez urządzenia typu Map-Server.

Niezaprzeczalną zaletą rozwiązania LISP-Mobility jest natomiast kompatybilność z bazowym rozwiązaniem LISP, a zatem również z obecnymi rozwiązaniami IP oraz związana z tym łatwość wdrożenia. Architektura LISP jest również rozwiązaniem elastycznym i łatwo rozszerzalnym, mogącym stanowić platformę dla większej liczby dodatkowych usług sieciowych.

### 8.3 MobilityFirst

O ile rozwiązania HIP i LISP można określić mianem ewolucyjnych w stosunku do MIP, to rozwiązanie MobilityFirst nosi znamiona propozycji rewolucyjnej. Projekt MobilityFist [42],[43] proponuje bowiem podejście, w którym przyjmuje

się mobilność wszystkich elementów sieci, a zatem obsługę wszystkich użytkowników zarówno stacjonarnych, jak i mobilnych w taki sam sposób.

MobilityFirst wprowadza szereg nowych założeń odnośnie sposobu funkcjonowania sieci. Do najistotniejszych z naszego punktu widzenia należą:

- Separacja identyfikatora obiektu i jego informacji adresowych (wykorzystanie GUID - Globally Unique ID),
- Zastosowanie nowego protokołu Global Name Resolution Service (GNRS) pozwalającego powiązać te odseparowane informacje,
- Zaproponowanie dedykowanego, hybrydowego protokołu routingu pozwalającego na tymczasowe buforowanie przesyłanych danych.

Przyjęte założenia prowadzą do braku powiązania identyfikatora obiektu z jego aktualną lokalizacją w strukturze sieci, jak też z topologią sieci w ogólności.

Przykładowa procedura przemieszczania się między dwiema podsieciami w architekturze MobilityFirst została zaprezentowana na Rys. 18.

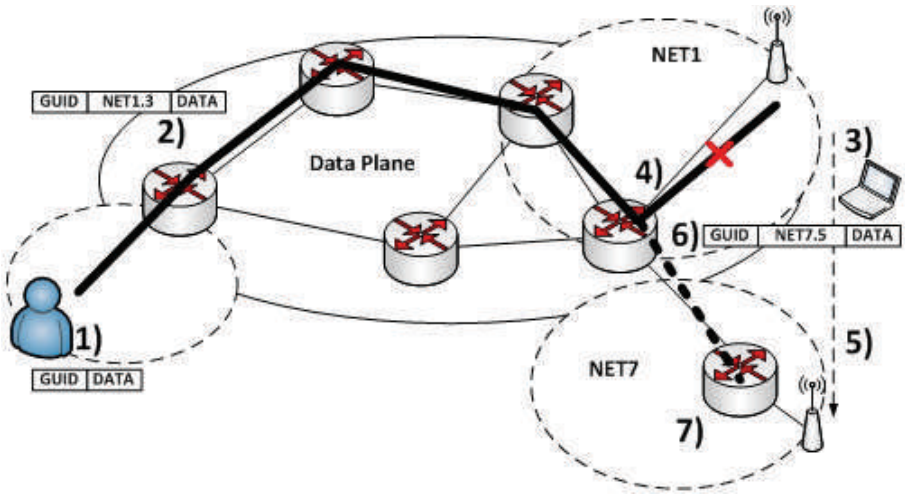
W celu nawiązania połączenia z innym elementem sieci albo uzyskania dostępu do zasobu użytkownik posługuje się wysokopoziomowym identyfikatorem. Identyfikator ten jest abstrakcyjnym określeniem docelowego podmiotu, który jest łatwy do zapamięta i wykorzystania. Przykładowo może być to opis tekstowy określający możliwości urządzeń użytkownika, z którym chcemy się skomunikować np. "wszystkie urządzenia pozwalające na rozmowę głosową użytkownika X".

Interpretacja tego identyfikatora oraz jego zamiana na globalny identyfikator GUID lub zestaw identyfikatorów (w przypadku komunikacji wielopunktowej) realizowana jest przy pomocy dedykowanych usług sieciowych (1).

GUID jest wewnętrznym identyfikatorem unikalnym w skali całej sieci pozwalającym na adresowanie obiektów, do których zaliczyć możemy zasoby (dane) i elementy (urządzenia) dostępne w sieci. Nie jest on jednak powiązany z topologią sieci i nie pozwala na określenie aktualnego punktu podłączenia obiektu. W MobilityFirst zaproponowano sposób tworzenia GUID oparty na funkcjach skrótu, pozwalających łatwo przekształcić go na identyfikator obszaru sieciowego (np. systemu autonomicznego), w którym przechowywane są szczegółowe informacje (w tym aktualny punkt podłączenia) na temat danego obiektu.

Na podstawie powyższych informacji GUID jest odwzorowywany na adres sieci transportowej, która wykorzystywana jest do samego przesyłania danych (2). Informacje adresowe sieci transportowej (np. adres IP, identyfikator sieci + port) zostają dołączone do nagłówka pakietu. Sieć transportowa może być wykorzystywać dowolne protokoły, a nawet ulegać zmianie bez dezorganizacji działania warstw wyższych - z tego względu można uznać MobilityFirst za rozwiązanie posiadające cechy rozwiązania wirtualizacyjnego. W celu zapewnienia odpowiedniej skalowalności możliwe jest wykorzystanie więcej niż jednej funkcji skrótu opisanej w poprzednim akapicie i przechowywania szczegółowych informacji o obiektach w wybranych obszarach sieci. Należy zauważyć, że mechanizmy sieci świadomych treści (Content Aware Network) znajdują w tym przypadku zastosowanie i pozwalają na dostęp do zasobów w sposób efektywny.

Zmiana punktu podłączenia z (3) na (5) powoduje zmianę lokalizatora (np. adresu IP) oraz uniemożliwia poprawne dostarczenie danych. Brak dostępu do obiektu powoduje uruchomienie procedury *late binding* w routerze brzegowym (4). Pozwala ona na ponowne odwzorowanie identyfikatora GUID na nowy adres sieciowy (lokalizator). Dodatkowo router brzegowy może zastosować procedurę buforowania przesyłanych danych, która minimalizuje liczbę straconych pakietów. Po rejestracji obiektu w nowej sieci, router uzyskuje jego adres (6) i może dokończyć transmisję danych zgodnie z nową ścieżką (7).

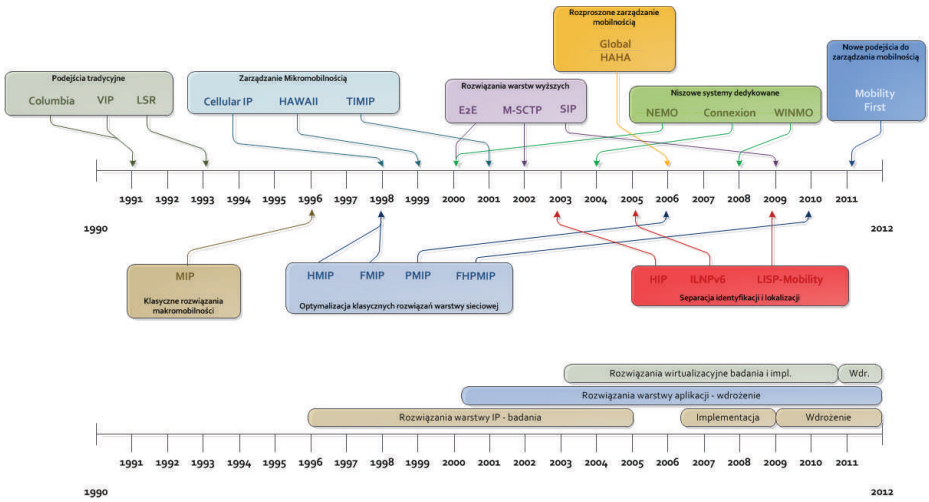



**Rysunek 18.** Przełączenie w architekturze MobilityFirst.

## 9 Podsumowanie

Początkowy rozwój rozwiązań zarządzania mobilnością przebiegał stosunkowo powoli (Rys. 19) - ponad 10 lat upłynęło od momentu opracowania rozwiązania MIP, do momentu, w którym doczekano się pierwszych implementacji odpowiednich do zastosowania na szeroką skalę w systemach produkcyjnych. Znacząca większość tego czasu poświęcona została badaniom teoretycznym i opracowywaniu efektywnych optymalizacji jego działania, trudnych jednakże w praktycznym wdrożeniu. Dopiero od stosunkowo niedawna rozwój protokołów zarządzania mobilnością w warstwie sieciowej przebiega w powiązaniu z rozwojem praktycznych systemów sieciowych.

Obserwowana obecnie konwergencja systemów i sieci teleinformatycznych prowadzi do ujednolicenia wykorzystywanych protokołów i mechanizmów. Zarządzanie mobilnością na poziomie sieci IP pozwala na integrację różnych, często




**Rysunek 19.** Rozwój (badania, implementacja i wdrożenia) zróżnicowanych grup protokołów zarządzania mobilnością.

heterogenicznych systemów i sieci, wpisuje się wiec ściśle w powyższą filozofię. W związku z możliwością współpracy protokołów warstwy sieciowej z wieloma technikami transmisji warstw niższych oraz brakiem ograniczeń dla sposobu realizacji usługi w warstwach wyższych, jest ona naturalnym miejscem ulokowania mechanizmów zarządzania mobilnością.

Kierunek rozwoju różnorodnych technik wykorzystywanych w sieciach komputerowych wskazuje, iż w najbliższej przyszłości jednorodnym protokołem wykorzystywanym w warstwie sieciowej pozostanie protokół IP. W pierwszej części rozdziału przedstawione zostały, między innymi, dwie główne grupy protokołów mobilności w warstwie IP - Mobile IP oraz Proxy Mobile IP. Posiadają one jednak znaczne ograniczenia wynikające z braku rozróżnienia identyfikatora i lokalizatora terminala mobilnego poprzez protokół IP, co przekłada się na efektywność przełączania. Sposoby ich optymalizacji oraz mechanizmy usprawnienia obsługi terminali mobilnych zaprezentowano w drugiej części rodziału.

Rozszerzając horyzont czasowy prowadzonych rozważań, można przypuszczać, że rozwiązania IP zostaną zastąpione nowymi protokołami, eliminującymi słabości powyższych rozwiązań w obsłudze urządzeń mobilnych.

Najnowszym trendem są rozwiązania oparte na różnorodnych technikach wirtualizacyjnych, w szczególności dążących do rozdzielenia identyfikacji adresata od jego lokalizacji w systemie sieciowym oraz do wprowadzenia separacji warstw wyższych od struktury tzw. sieci transportowej. Rozwój tego nurtu rozwiązań cechuje się równoległymi pracami badawczo rozwojowymi i implementacyjnymi w niewielkich środowiskach sieciowych. Wyniki prac osiągają właśnie poziom po-

zwalający na przeprowadzenie udanych wdrożeń na skalę globalnych systemów Internetowych.

Przykładowe propozycje takich architektur - HIP, LISP oraz MobilityFirst zostały opisane w drugiej części rozdziału, jako rozwiązania pozwalające na zarządzania użytkownikami mobilnymi w Internecie Przyszłości.

Obserwując rozwój i obecny, zaawansowany stan badań oraz implementacji rozwiązań zarządzania mobilnością w warstwie sieciowej, należy jednak pamiętać, iż efektywna obsługa użytkowników mobilnych wymaga również odpowiedniej obsługi w warstwach niższych modelu ISO-OSI. Minimalizacja zakłóceń łączności związanych ze słabą jakością sygnału radiowego oraz procesem przełączenia pomiędzy punktami dostępu do sieci jest wymogiem niezbędnym do realizacji nowoczesnych usług przeznaczonych dla użytkowników mobilnych. Na szczęście można zaryzykować stwierdzenie, iż badania oraz implementacja mechanizmów związanych z tym aspektem obsługi mobilności prowadzone są obecnie z równym nasileniem, jak ma to miejsce w przypadku opisywanych w powyższym artykule mechanizmów warstw wyższych. Pozwala to oczekiwać znaczącego upowszechnienia kompleksowych rozwiązań zarządzania mobilnością w popularnych systemach sieciowych i związanego z tym dalszego, intensywnego rozwoju usług dostępnych, lub wręcz dedykowanych, dla użytkowników mobilnych.

## Literatura

1. Gustafsson E., Jonsson A.: Always best connected, Wireless Communications, IEEE, vol.10, no.1, pp. 49- 55, February 2003
2. Bagnulo, M., Garcia-Martinez, A., Azcorra, A.: IPv6 multihoming support in the mobile internet, Wireless Communications, IEEE, vol.14, no.5, pp.92-98, October 2007
3. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011-2016, Cisco White Paper, 2012
4. ITU-T Recommendation Y.1541: Network performance objectives for IP-based Services, December 2011
5. ITU-T Recommendation G.114: One-way Transmission Tim, May 2012
6. Shrestha G. M., Jasperneite J.: Performance Evaluation of Cellular Communication Systems for M2M Communication in Smart Grid Applications, (Kwiecien A. et al. Eds) CN2011, CCIS 191. Springer Verlag 2012, s 352-359
7. 3GPP - LTE: http://www.3gpp.org/LTE/ (dostępna 21.07.2012)
8. Machań P.: Effectiveness of IP Traffic Handover in IEEE 802.11 Wireless Local Area Networks. Ph.D. Dissertation, Gdańsk 2012
9. Le D., Fu X., Hogrefe D.: A Review of Mobility Support Paradigms for the Internet. Tech. Rep. Georg-August-Universität Göttingen, Institut für Informatik. 2005
10. Fikouras, N.A. et al.: Performance Evaluation of TCP over Mobile IP, 2001
11. Mogul J., Deering S.: Path MTU Discovery, IETF RFC 1191, November 1990
12. Jung H., Koh S. J.: Problem Statements and Requirements for Mobile Oriented Future Internet, International Conference on ICT Convergence (ICTC), September 2011
13. Zhu Z., Wakikawa R., Zhang L.: A Survey of Mobility Support in the Internet, IETF RFC 6301 - Informational, July 2011

14. Kempf J.: Problem Statement for Network-Based Localized Mobility Management (NETLMM), IETF RFC 4830, April 2007
15. Perkins C.: IP Mobility Support. RFC 2002 Work in Progress, October 1996
16. Perkins C.: IP Mobility Support for IPv4, RFC 3344, August 2002
17. Perkins C.: IP Mobility Support for IPv4, Revised. RFC 5944, November 2010
18. Johnson D., Perkins C., Arkko J.: Mobility Support in IPv6, RFC 3775, June 2004
19. Qing Li, Tatuya Jinmei, Keiichi Shima: Mobile IPv6: Protocols and Implementation, Morgan Kaufman, 2009.
20. Perkins C., Johnson D., Arkko J.: Mobility Support in IPv6, RFC 6275, July 2011
21. Gundavelli S., et al.: Proxy Mobile IPv6. RFC 5213, August 2008
22. Mysore, J. and V. Bharghavan, "A New Multicast-based Architecture for Internet Host Mobility", ACM Mobicom, 1997
23. Andrew, L., "A Border Gateway Protocol 4 (BGP-4)", Boeing White Paper, 2006
24. Wozniak J., et al.: Comparative analysis of IP-Based Mobility Protocols and Fast Handover Algorithms in IEEE 802.11 based WLANS. (Kwiecien A. et al. Eds): CN2011, CCIS 160. Springer Verlag, pp. 87-101, 2011
25. Campbell A., et al.: Cellular IP, Intemet Draft, draft-ietf-mobileip-cellularip-00, Work in Progress, 1999
26. Ramjee R., Varadhan K., Salgarelli L.: HAWAII: A Domain-based Approach for Supporting Mobility in Wide-area Wireless Networks, IEEE/ACM Transcations on Networking, 2002
27. Soliman H., et al.: Hierarchical Mobile IPv6 (HMIPv6) Mobility Management. RFC 5380, 2008
28. Koodli R. (ed.): Mobile IPv6 Fast Handovers, RFC 5568, IETF Proposed Standard, 2009
29. Koodli R., Perkins C.: Mobile IPv4 Fast Handovers, RFC 4988, 2007
30. Yokota H., et al.: Fast handovers for PMIPv6, RFC 5949, IETF Proposed Standard, 2010
31. Wakikawa R., Valadon G., Murai J.: Migrating Home Agents Towards Internet-scale Mobility Deployment, ACM CoNEXT, 2006
32. Jung H., Koh S.: Distributed Mobility Control in Proxy Mobile IPv6 Networks, IEICE Transactions on Communications, Vol. E94-B, No. 8, pp. 2216 - 2224, 2011
33. Singh R., Teo Y. C., Yeow S. W.: RAT: A Quick (and Dirty) Push for Mobility Support. Proc. 2nd IEEE Workshop on Mobile Computer Systems and Applications, 1999
34. Buddhikot M., et al.: MobileNAT: A New Technique for Mobility Across Heterogeneous Address Spaces. MONET, vol. 10, No. 3, pp. 289-302, 2005
35. Chen X., Kriaras I., Paparella A.: Non-encapsulation mobile IP, U.S. Patent No. 6,842,456, European Patent No. EP 1032178, 2005
36. Snoeren A., Balakrishnan H.: End-to-End Approach to Host Mobility. ACM Mobicom, 2000
37. Rosenberg J., et al.: SIP: Session Initiation Protocol, RFC 3261, IETF Proposed Standard, 2002
38. Schulzrinne H., Wedlund E.: Application-Layer Mobility Using SIP. Mobile Computing and Communications Review, 2010
39. Moskowitz R., et al.: Host Identity Protocol, IETF RFC 5201, 2008
40. Farinacci, D., et al., "LISP Mobile Node", IETF Internet Draft, 2012
41. Farinacci, D., et al., "Locator/ID Separation Protocol (LISP)", IETF Internet Draft, 2011
42. Homepage of MobilityFirst project http://mobilityfirst.winlab.rutgers.edu/
43. Seskar I., et al.: MobilityFirst Future Internet Architecture Project, 6th International Conference on Future Internet Technologies, Seoul, Korea, 2011

# Dobre praktyki automatyzacji procesu migracji do protokołu IPv6

Krzysztof Nowicki, Wojciech Gumiński, Paweł Kamiński, Łukasz Łoska, Karol Podolski, Krzysztof Wasilewski

Katedra Teleinformatyki, Politechnika Gdańska

**Streszczenie** W pracy przedstawiono możliwość realizacji automatów migracji do IPv6 systemów, usług i aplikacji. Dokonano klasyfikacji metod migracji ze względu na możliwość ich automatyzacji. Opisano założenia i sposób implementacji wybranych narzędzi wspomagających administratora sieci w realizacji procesu migracji systemów operacyjnych, urządzeń wbudowanych jak też usług i aplikacji sieci do protokołu IPv6. Zaprezentowano projekt i implementacje oprogramowania integrującego narzędzia wspierające automatyczną migrację do IPv6.

## 1 Wprowadzenie

Różnorodność systemów operacyjnych, sprzętu i metod dostępu do sieci IPv6 [2] skutkuje potrzebą wprowadzenia dobrych praktyk w zakresie migracji do protokołu IPv6 [3-6]. Migracja z IPv4 do IPv6 nie jest trywialna. Z powodu dużej liczby i złożoności elementów, z których składają się obecne sieci, zagadnienie przeniesienia komunikacji ze starszej na nowszą technologię IP jest skomplikowane [7]. Aby dokonać migracji sieci z IPv4 do IPv6 trzeba zmienić konfigurację wielu elementów realizujących komunikację w danej sieci. Elementami tymi są: urządzenia sieciowe, systemy operacyjne, usługi sieciowe i aplikacje sieciowe. Taką zmianę trzeba również przeprowadzić w odpowiedniej kolejności. Najpierw należy dokonać zmian konfiguracji na urządzeniach sieciowych, by możliwa była komunikacja IPv6 w sieci. Następnie należy przystosować systemy operacyjne węzłów w sieci, by obsługiwały połączenia po IPv6. W dalszej kolejności należy zmigrować usługi sieciowe poprzez zmiany konfiguracji poszczególnych aplikacji sieciowych.

Jedną z propozycji ułatwiającej administratorom rozwiązanie problemu migracji jest przygotowywana w ramach projektu Inżynieria Internetu Przyszłości [8] aplikacja „Przewodnik migracji do IPv6" [9], która ma na celu ułatwienie tego procesu. Przewodnik migracji przedstawia proces migracji z IPv4 do IPv6 według wyżej omawianej koncepcji. Przewodnik migracji jest źródłem wiedzy o protokole IPv6. Zawiera on zbiór porad w zakresie dobrych praktyk wdrażania protokołu IPv6 w sieciach [10-11]. Został opracowany ze szczególnym skupieniem na sieciach małych, typu SOHO (ang. Small Office Home Office), w przypadku których osoba zarządzająca siecią (administrator) nie zawsze posiada ugruntowaną wiedzę teleinformatyczną [12]. Koncepcja przewodnika oparta jest na

założeniu, że administrator uzyska poradę jak ma zmigrować sieć, a na podstawie porad samodzielnie, wg wskazówek, dokona procesów rekonfiguracyjnych urządzeń i, w efekcie, migracji sieci.

Przewodnik nie zwalnia jednak administratora od uciążliwych czynności konfiguracyjnych, zatem niezwykle użyteczne byłoby proces migracji, w możliwie jak największym stopniu, zautomatyzować.

Automatyzacja procesu migracji (niezależnie czy to migracja urządzeń sieciowych, systemów operacyjnych, usług, czy też aplikacji) może być pożądana z wielu powodów, ponieważ oznacza bezobsługowe, samoczynne wykonywanie niezbędnych czynności potrzebnych do umożliwienia komunikacji w sieci IPv6. W przypadku, gdy występuje pewna powtarzalność takich czynności, automatyzacja powoduje usprawnienie procesu migracji. Zapewnia, że wykonywane są wszystkie konieczne akcje w określonej kolejności. Czasami nieostrożny administrator może mieć z tym problem. Ponadto automatyzacja eliminuje błędy powstałe poprzez literówki lub niedokładne przepisanie komend. Dla długich i „nieprzyjaznych" człowiekowi adresów IPv6 może mieć to szczególne znaczenie. System automatycznej migracji IPv4 do IPv6 może okazać się niezbędnym narzędziem w sytuacji konieczności wykonywania tych samych czynności na wielu urządzeniach sieciowych, systemach operacyjnych czy też przy wielu aplikacjach. Przykładem są rozbudowane centra informatyczne, gdzie istnieje wiele serwerów o identycznej platformie systemowej oraz analogicznie skonfigurowanych aplikacjach (tzw. farmy serwerów). System automatycznej migracji aplikacji z IPv4 do IPv6 mógłby zarówno uruchomić obsługę protokołu IPv6 np. dla aplikacji jak i odpowiednio ją skonfigurować. Czas potrzebny na przeprowadzenie procesu migracji w takich sytuacjach znacząco się zmniejsza, co również jest niewątpliwą zaletą.

Jednym z powodów wolniejszej, niż było to zakładane, migracji sieci IPv4 do IPv6 jest brak wiedzy o IPv6 wśród administratorów, szczególnie mniejszych sieci [5]. Brak wiedzy wynika z faktu, że z jednej strony na rynku nie jest oferowanych zbyt wiele szkoleń w zakresie migracji do IPv6, a z drugiej strony oferta zwartych materiałów, które opisywałyby te procesy dla konkretnych rozwiązań sieci IPv4 jest znikoma [7]. Brakuje również „tanich specjalistów", którzy świadczyliby tego rodzaje usług dla małych sieci, w tym sieci domowych [12].

Odbiorcami, czyli osobami wykorzystującymi system automatycznej migracji IPv4 do IPv6, będą administratorzy, którzy niekoniecznie muszą posiadać pełną wiedzę odnośnie praktyk przenoszenia komunikacji do sieci IPv6. Wystarczy, że zapoznają się z informacjami na temat obsługi takiego systemu. W wypadku np. systemów automatycznej migracji aplikacji z IPv4 do IPv6, administratorzy aplikacji nie będą musieli dysponować szeroką wiedzą odnośnie przenoszenia funkcjonowania aplikacji do IPv6. System automatycznej migracji aplikacji powinien wykonać szereg czynności na podstawie ewentualnych wskazówek administratora aplikacji, posługującego się takim systemem.

## 2 Metody automatyzacji procesu migracji

Migracja do obsługi protokołu IPv6 zwykle obejmuje konfigurację dostępu do sieci, konfigurację podstawowych usług sieciowych oraz konfigurację aplikacji korzystających z połączeń sieciowych. Automatyzację procesu migracji konfiguracji do obsługi protokołu IPv6 można przeprowadzić na kilka różnych sposobów.

Metoda pierwsza (mimesis) polega na naśladowaniu czynności wykonywanych przez użytkownika. Za pomocą skryptów umożliwiających symulacje zachowania człowieka przy konsoli, np. wykorzystując pakiet Expect [13], można dokonać niezbędnych zmian w konfiguracji systemu. Ograniczeniem tej metody jest konieczność instalowania skryptów i często również dodatkowych pakietów oprogramowania w systemie, co może być niemożliwe, w szczególności dla systemów wbudowanych.

Metoda druga (przetwarzania plików konfiguracyjnych) wykorzystuje spostrzeżenie, że większość systemów przechowuje albo umożliwia wykonanie eksportu i importu informacji o aktualnej konfiguracji systemu w postaci plików tekstowych. Możliwa jest zatem migracja konfiguracji za pomocą zewnętrznej aplikacji dokonującej analizy i przetworzenia pliku konfiguracyjnego. Migracja konfiguracji wykonywana tym sposobem wykonywana jest w środowisku niezależnym od migrowanego systemu, co pozwala uczynić ją bardziej uniwersalną i rozbudowywalną o możliwość migracji kolejnych systemów. Dodatkową zaleta migracji wykonywanej metodą przetwarzania plików konfiguracyjnych jest pozostawienie administratorowi możliwości przejrzenia i dokonania ręcznych poprawek w wynikowym pliku konfiguracyjnym przed jego zastosowaniem w docelowym systemie

## 3 Migracja do IPv6 konfiguracji dostępu, usług oraz aplikacji

**Migracja konfiguracji dostępu do sieci IPv6** obejmuje przypisanie właściwej adresacji IPv6 do interfejsów sieciowych, skonfigurowanie trasy domyślnej oraz adresów serwerów DNS (ang. *Domain Name Server*). Dla systemów, które nie są bezpośrednio podłączone do sieci IPv6, dochodzi konfiguracja właściwego tunelu 6in4 (6to4) dla systemów posiadających własny adres IPv4 albo tunelu Teredo [14] dla systemów znajdujących się za urządzeniem z translacją adresów NAT (ang. *Network Address Translation*). Do wykonania tych czynności konieczna jest znajomość aktualnej adresacji IPv4 systemu oraz docelowej adresacji IPv6. Dla systemów klienckich posługujących się tunelami 6to4 oraz Teredo adresacja IPv6 jest wyznaczana automatyczne, a dla systemów z bezpośrednim dostępem do sieci IPv6 pochodzi ona od operatora i musi być znana w czasie przeprowadzania migracji.

Za **podstawowe usługi sieciowe** niezbędne do prawidłowego działania sieci IPv6 uznajemy tu usługi rozgłaszania prefiksu sieci RA (ang. *Router Advertisement*) dla bezstanowej konfiguracji klientów sieci SLAAC (ang. *Stateless Address Autoconfiguration*), usługę serwera automatycznej konfiguracji hostów DHCPv6

(ang. *Dynamic Host Configuration Protocol*) oraz usługę pośredniczącego rekursywnego serwera DNS [15, 16]. Konfiguracja tych usług sprowadza się w większości systemów do uruchomienia ich na określonych interfejsach sieciowych z zadaniem parametrów, które wynikają z adresacji interfejsów. W sieciach, w których używany jest serwer DHCPv6 należy pamiętać o ustawieniu flagi Managed w konfiguracji usługi RA.

Stosowane powszechnie protokoły routingu dynamicznego RIP, BGP, OSPF oraz EIGRP dla routerów Cisco posiadają obsługę protokołu IPv6. Konfiguracja ich do pracy w sieciach IPv6 jest bardziej skomplikowana. Dla routingu statycznego oraz protokołu BGP konfiguracja jest bardzo zbliżona do konfiguracji dla IPv4. Dla protokołów RIP, OSPF i EIGRP oprócz adresacji konieczna jest znajomość właściwych interfejsów, co nie zawsze w sposób oczywisty wynika z konfiguracji IPv4. Również zaawansowane parametry służące do „dostrajania" protokołów routingu dynamicznego różnią się czasami nazewnictwem w zakresie protokołów IPv4 i IPv6.

**Migracja polityk bezpieczeństwa** realizowanych za pomocą zapory (ang. *Firewall*) oraz list kontroli dostępu ACL (ang. *Access Control List*) jest zagadnieniem najtrudniejszym w procesie automatyzacji. Trudność automatyzacji procesu migracji zasad dostępu do sieci wynika z konieczności posługiwania się tu adresacją systemów zewnętrznych, która w żadnym stopniu nie jest zależna od konfiguracji adresów migrowanego systemu. Automatyzacja może się tu sprowadzić do przeniesienia reguł oczywistych i podpowiedzi w przypadku reguł niezwiązanych jednoznacznie z konfiguracją migrowanego sytemu. Jest to kolejny przykład potwierdzający konieczność i wygodę stosowania migracji plików konfiguracyjnych.

**Migrator aplikacji** wykrywa ustawienia systemu operacyjnego oraz aplikacji i dokonuje odpowiednich zmian w plikach konfiguracyjnych aplikacji. Możliwość implementacji systemu automatycznej migracji aplikacji z IPv4 do IPv6 zależy w głównej mierze od obsługi IPv6 danej aplikacji. Tabela 1 przedstawia podsumowanie obsługi IPv6 zbadanych aplikacji.

| **Nazwa aplikacji** | **Obsługa IPv6** | **Od kiedy?** |
|---|---|---|
| Apache HTTP Server | jest | Styczeń 2004 |
| NginX HTTP Server | jest | 21 lutego 2009 |
| ProFTPD | jest | 11 września 2003 |
| vsftpd | jest* | 2003 |
| PostgreSQL | jest | 17 listopada 2003 |
| MySQL | brak | - |
| FreeRADIUS | jest* | 16 maja 2007 |
| ISC DHCP | jest* | 20 lipca 2007 |

**Tablica 1.** Zestawienie wybranych aplikacji pod kątem obsługi IPv6.

* aplikacje, które mogą przyjmować połączenia albo IPv4 albo IPv6.

Jak widać nie wszystkie aplikacje posiadają obsługę IPv6. Z oczywistych względów nie da się ich zmigrować. Niektóre aplikacje posiadają obsługę IPv6, ale nie mogą jednocześnie działać w oparciu o IPv4 i IPv6 – potrzebne jest skonfigurowanie oddzielnych instancji takich aplikacji (w tabeli 1 zaznaczono takie aplikacje gwiazdką). W pozostałych przypadkach aplikacje posiadają obsługę IPv6 i mogą jednocześnie działać w jednej i drugiej technologii IP.

Warto też zauważyć, że pełne obsługa IPv6 niektórych aplikacji, nastąpiło zaledwie kilka lat temu. Stąd też dopiero od kilku lat stało się możliwe przejście Internetu na nowy protokół, a nie jak często twierdzi się, już pod koniec XX wieku. Mimo to, nadal nie wszystkie aplikacje są przygotowane do pracy w sieciach IPv6.

Nie wszystkie więc działania migracyjne można całkowicie zautomatyzować. Nie jest to możliwe w szczególności, gdy konieczne jest posiadanie wiedzy o środowisku zewnętrznym np. przydzielonej puli adresów. Migracji takiej nie można również przeprowadzać w sposób całkowicie automatyczny, ponieważ nie stosuje się bezpośredniego i jednoznacznego przyporządkowywania adresów IPv4 i IPv6. Szczegółowe parametry konfiguracji poszczególnych usług sieciowych mają odmienne wartości dla protokołów IPv4 i IPv6.

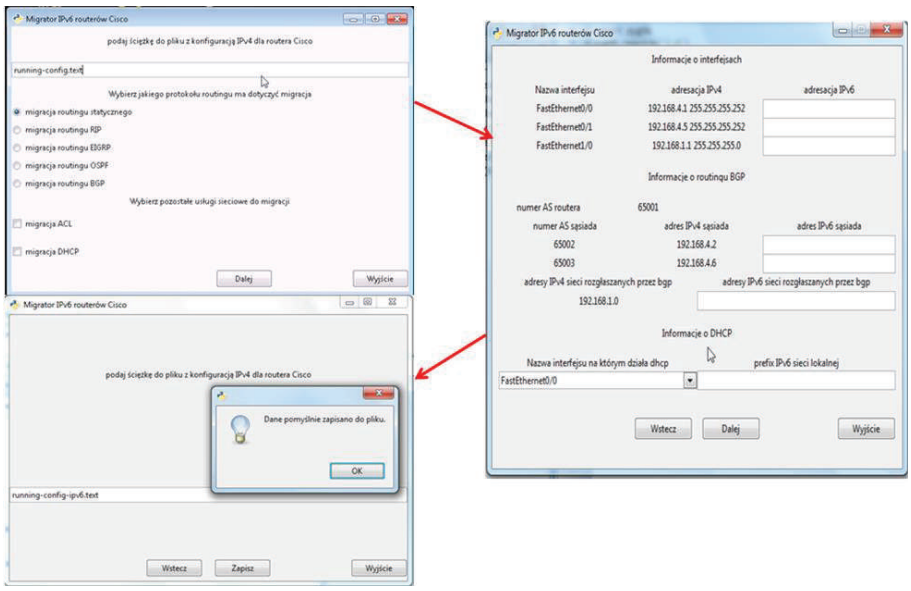
Dla potrzeb migracji konfiguracji niezbędna jest znajomość docelowej adresacji IPv6 oraz parametrów usług sieciowych niezależnych od migrowanego systemu jak np. numeracja systemów autonomicznych albo obszarów dla routingu dynamicznego BGP i OSPF. Wymusza to zwykle zastosowanie migracji, w której niezbędny jest udział człowieka w celu podania wymaganych wartości parametrów konfiguracyjnych.

## 4 Przykłady migratorów do sieci IPv6

W pracach nad migracją sieci z IPv4 do IPv6, prowadzonych w ramach grantu Inżynieria Internetu Przyszłości, zrealizowano szereg systemów automatycznej migracji opartych o metodę przetwarzania plików konfiguracyjnych. Zrealizowane aplikacje dokonujące automatycznej migracji plików konfiguracyjnych do obsługi protokołu IPv6 potwierdziły słuszność poczynionych założeń oraz wykazały przydatność automatyzacji procesu migracji do IPv6.

### 4.1 Realizacja migratora konfiguracji dostępu do sieci IPv6

Interfejs aplikacji dokonującej automatyczną migrację plików konfiguracyjnych do obsługi protokołu IPv6 przedstawiano na rysunku 1. W chwili obecnej aplikacja obsługuje migrację plików konfiguracyjnych routerów sprzętowych Cisco [17], routerów wirtualnych Vyatta [18] oraz routerów klasy SOHO z systemem OpenWRT [19]. Aplikacja umożliwia migrację konfiguracji interfejsów, routingu statycznego, usług RA, DHCP i DNS, protokołów routingu dynamicznego RIP, BGP, OSPF i EIGRP oraz w ograniczonym zakresie zapory ip6tables dla OpenWRT i list ACL dla systemów Cisco i Vyatta. Zarówno istniejąca konfiguracja IPv4 jak i docelowa konfiguracja obejmująca obsługę IPv6 jest z sys-



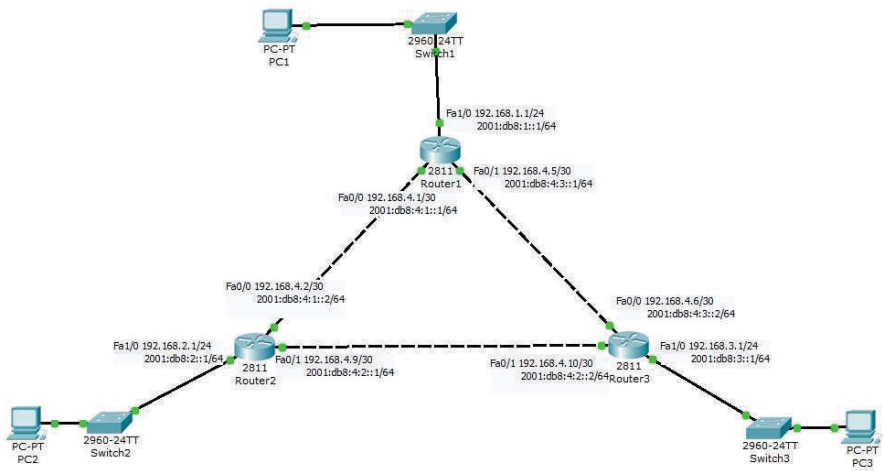

**Rysunek 1.** Aplikacja automatyzacji procesu migracji konfiguracji do IPv6

temu pobierana i zapisywana w nim automatycznie przy wykorzystaniu dostępu zdalnego z wykorzystaniem protokołów SSH albo Telnet.

Aplikacja automatyzacji procesu migracji konfiguracji do IPv6 była testowana w środowisku homo i heterogenicznym. Konfigurację środowiska testowego przedstawiono na rysunku 2. Dla wszystkich opisywanych systemów Cisco, Vyatta, OpenWRT i wszystkich protokołów routingu dynamicznego uzyskano w pełni działającą sieć dwuprotokolarną IPv4 i IPv6 bazując wyłącznie na konfiguracji IPv4 routerów i danych dotyczących adresacji IPv6 wprowadzonych do aplikacji automatyzacji procesu migracji konfiguracji.

### 4.2 Realizacja systemu automatycznej migracji aplikacji

Przy realizacji systemu automatycznej migracji aplikacji sieciowych [20] Apache HTTP Server [21] oraz ProFTPD [22] z IPv4 do IPv6 przyjęto, że musi on wspierać dwa tryby migracji: *Dual Stack* i *Native IPv6*. Migrator po wykryciu ustawień systemu operacyjnego oraz aplikacji dokonuje się odpowiednich zmian w plikach konfiguracyjnych aplikacji. Najpierw system automatycznej migracji sprawdza, w jakiej wersji zainstalowana jest aplikacja, która ma być zmigrowana. Bada również, czy aplikacja została zainstalowana wraz ze obsługą IPv6 (czasem podczas instalacji aplikacji można zrezygnować z takiej obsługi). Następnie migrator aplikacji sprawdza wszystkie interfejsy sieciowe dostępne w systemie operacyjnym w poszukiwaniu globalnych adresów IPv6. Zapamiętuje, jakie adresy IPv4 są przypisane do takich interfejsów. W kolejnym kroku migrator aplikacji




**Rysunek 2.** Konfiguracja środowiska testowego procesu migracji do IPv6

odnajduje lokalizację plików konfiguracyjnych i w zależności od ich umiejscowienia i formatu dokonuje stosownych zmian w plikach konfiguracyjnych aplikacji. Dla migracji *Dual Stack* zmiany w plikach konfiguracyjnych aplikacji polegają na dodaniu obok istniejących wpisów IPv4 analogicznych wpisów dla IPv6 (odbieranie połączeń na adresach IPv6, konfiguracja hostów wirtualnych dla IPv6). Natomiast przy migracji *Native IPv6* wszelkie wpisy z adresami IPv4 są zamieniane na adresy IPv6.

Migrator jest dedykowany dla konkretnej aplikacji – oddzielny dla Apache HTTP Server i oddzielny dla ProFTPD. Dzięki temu poszczególne programy nie muszą być rozbudowane, ich kod jest bardziej przejrzysty i nie trzeba podawać dodatkowych parametrów związanych z wskazywaniem aplikacji, której konfiguracja podlega migracji podczas uruchamiania programu [20]. Do realizacji migratora wykorzystano bashowe skrypty wspomagane interpretowanym językiem programowania AWK.

Migrator aplikacji Apache HTTP Server oraz ProFTPD zbudowany według określonych w niniejszym rozdziale założeń, z powodzeniem realizuje postawione przed nim zadania. Dokonuje migracji aplikacje zarówno w oparciu tryb *Dual Stack* jak i *Native IPv6*. Analizuje istniejącą konfigurację systemu i aplikacji (w tym czy aplikacja posiada skompilowaną obsługę dla IPv6). W dalszej części przedstawiono koncepcje ich realizacji i uzyskane wyniki na przykładzie migratora aplikacji Apache HTTP Server. W zależności od wyników i podanego parametru ( *Dual Stack* – patrz rys. 3, lub *nativev6* – patrz rys. 4) dokonuje stosownych zmian [20].

Istotną cechą realizowanego migratora aplikacji jest zapewnienie odwracalności procesu migracji (patrz rysunek 5). Możliwe jest również uruchomienie migra-

```
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 30.0.2.15:80                0.0.0.0:*                   LISTEN      1503/httpd
tcp        0      0 20.0.2.15:80                0.0.0.0:*                   LISTEN      1503/httpd
root@vinfin-sl.ds.pg.gda.pl:~# apache_migrator.sh dualstack
Wersja Apache 2.0 lub wyższa                     [TAK]
Apache ze wsparciem IPv6                         [TAK]

Interfejs eth0 nie ma globalnego adresu IPv6
Interfejs eth1 ma globalny adres IPv6:           31de::a00:2ff:fe31:3131
Interfejs eth2 ma globalny adres IPv6:           31de::a00:3ff:fe31:3131

Znaleziono lokalizację plików konfiguracyjnych:  /etc/httpd

Wykonywanie kopii plików konfiguracyjnych do /etc/httpd/v4back:
razem 36K
-rw-r--r--. 1 root root 35K 05-28 19:36 httpd.conf

Zmiana nasłuchiwania z IPv4 na IPv6 - tryb dualstack

Zatrzymywanie httpd:                                        [  OK  ]
Uruchamianie httpd:                                         [  OK  ]
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 30.0.2.15:80                0.0.0.0:*                   LISTEN      1629/httpd
tcp        0      0 20.0.2.15:80                0.0.0.0:*                   LISTEN      1629/httpd
tcp        0      0 31de::a00:3ff:fe31:3131:80  :::*                        LISTEN      1629/httpd
tcp        0      0 31de::a00:2ff:fe31:3131:80  :::*                        LISTEN      1629/httpd
```

**Rysunek 3.** Migracja aplikacji Apache HTTP Server w trybie Dual Stack

```
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 30.0.2.15:80                0.0.0.0:*                   LISTEN      1708/httpd
tcp        0      0 20.0.2.15:80                0.0.0.0:*                   LISTEN      1708/httpd
root@vinfin-sl.ds.pg.gda.pl:~# apache_migrator.sh nativev6
Wersja Apache 2.0 lub wyższa                     [TAK]
Apache ze wsparciem IPv6                         [TAK]

Interfejs eth0 nie ma globalnego adresu IPv6
Interfejs eth1 ma globalny adres IPv6:           31de::a00:2ff:fe31:3131
Interfejs eth2 ma globalny adres IPv6:           31de::a00:3ff:fe31:3131

Znaleziono lokalizację plików konfiguracyjnych:  /etc/httpd

Wykonywanie kopii plików konfiguracyjnych do /etc/httpd/v4back:
razem 36K
-rw-r--r--. 1 root root 35K 05-28 19:44 httpd.conf

Zmiana nasłuchiwania z IPv4 na IPv6 - tryb nativev6

Zatrzymywanie httpd:                                        [  OK  ]
Uruchamianie httpd:                                         [  OK  ]
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 31de::a00:3ff:fe31:3131:80  :::*                        LISTEN      1814/httpd
tcp        0      0 31de::a00:2ff:fe31:3131:80  :::*                        LISTEN      1814/httpd
```

**Rysunek 4.** Migracja aplikacji Apache HTTP Server w trybie Native IPv6

tora w trybie testowym w celach diagnostycznych (patrz rysunek 6). Umożliwia to przeprowadzenie analizy migrowanej aplikacji bez wprowadzania zmian w jej konfiguracji. Rys 5. ukazuje odwracalność procesu automatycznej migracji uzyskiwaną poprzez uruchomienie migratora z parametrem *reverse*). Po dokonaniu zmian w konfiguracji migrator restartuje aplikację by zmiany zostały faktycznie wprowadzone. Rys. 6. przedstawia uruchamianie migratora w trybie testowym, w którym nie są wykonywane zmiany (parametr *check*).

```
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 31de::a00:3ff:fe31:3131:80  :::*                    LISTEN      2103/httpd
tcp        0      0 31de::a00:2ff:fe31:3131:80  :::*                    LISTEN      2103/httpd
root@vinfin-sl.ds.pg.gda.pl:~# apache_migrator.sh reverse
Wersja Apache 2.0 lub wyższa                   [TAK]
Apache ze wsparciem IPv6                       [TAK]

Interfejs eth0 nie ma globalnego adresu IPv6
Interfejs eth1 ma globalny adres IPv6:          31de::a00:2ff:fe31:3131
Interfejs eth2 ma globalny adres IPv6:          31de::a00:3ff:fe31:3131

Znaleziono lokalizację plików konfiguracyjnych:  /etc/httpd

Przywracanie konfiguracji IPv4 z /etc/httpd/v4back:
razem 36K
-rw-r--r--. 1 root root 35K 05-28 20:02 httpd.conf

Zatrzymywanie httpd:                                       [  OK  ]
Uruchamianie httpd:                                        [  OK  ]
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 30.0.2.15:80                0.0.0.0:*                    LISTEN      2183/httpd
tcp        0      0 20.0.2.15:80                0.0.0.0:*                    LISTEN      2183/httpd
```

**Rysunek 5.** Cofnięcie migracji aplikacji Apache HTTP Server z IPv6 do IPv4

```
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 30.0.2.15:80                0.0.0.0:*                    LISTEN      1947/httpd
tcp        0      0 20.0.2.15:80                0.0.0.0:*                    LISTEN      1947/httpd
root@vinfin-sl.ds.pg.gda.pl:~# apache_migrator.sh check
Wersja Apache 2.0 lub wyższa                   [TAK]
Apache ze wsparciem IPv6                       [TAK]

Interfejs eth0 nie ma globalnego adresu IPv6
Interfejs eth1 ma globalny adres IPv6:          31de::a00:2ff:fe31:3131
Interfejs eth2 ma globalny adres IPv6:          31de::a00:3ff:fe31:3131

Znaleziono lokalizację plików konfiguracyjnych:  /etc/httpd
root@vinfin-sl.ds.pg.gda.pl:~# netstat -tnpl | grep httpd
tcp        0      0 30.0.2.15:80                0.0.0.0:*                    LISTEN      1947/httpd
tcp        0      0 20.0.2.15:80                0.0.0.0:*                    LISTEN      1947/httpd
```

**Rysunek 6.** Uruchomienie trybu testowego migratora w celu sprawdzenia możliwości zmigrowania aplikacji Apache HTTP Server

Automatyczna migracja konfiguracji aplikacji Apache HTTP Server przebiega poprawnie niezależnie od adresów przypisanych do poszczególnych interfejsów sieciowych oraz niezależnie od zdefiniowanego w konfiguracji numeru portu dla protokołu HTTP.

## 5 Automat migracji dla konfiguracji usług i aplikacji sieciowych

W czasie trwania projektu Inżynieria Internetu Przyszłości [8] w Katedrze Teleinformatyki Politechniki Gdańskiej powstało kilka niezależnych aplikacji służących migracji konfiguracji z protokołu IPv4 do protokołu IPv6: opisanych w rozdziale 4 migracji konfiguracji dostępu do sieci IPv6 i migracji wybranych aplikacji sieciowych jak też usług DNS czy DHCP [23]. Oczywiście posługiwanie się odmiennymi, w zależności od realizowanego procesu migracji, aplikacjami jest

uciążliwe i czasochłonne. Stąd postanowiono opracować narzędzie integrujące (zwane dalej Automatem Migracji - AM) już powstałe aplikacje wspierające migracje jak tez umożliwiające łatwe dołączanie nowych. Automat Migracji ma na celu połączenie powstałych już migratorów, dalej zwanych skryptami, w jeden program, który będzie w stanie przeprowadzać migrację wielu usług i aplikacji sieciowych.

### 5.1 Projekt Automatu Migracji

W celu zrealizowania postawionego zadania, zostały przyjęte określone założenia i wytyczne dotyczące realizacji AM. Graficzny interfejs użytkownika został ujednolicony dla wszystkich skryptów migracji. Ułatwia to w znacznym stopniu korzystanie z programu. Komunikacja pomiędzy urządzeniami, a automatem odbywa się za pomocą protokołów ssh i/lub telnet. Wymiana danych pomiędzy migratorami (skryptami), a aplikacją AM została zrealizowana za pomocą plików xml. Jest to zabieg pozwalający zdefiniować dość uniwersalny interfejs dla różnych wymaganych do procesu migracji parametrów. Aplikacja AM posiada możliwość rozszerzania puli migratorów (puli skryptów) w prosty sposób poprzez dodanie kolejnych skryptów migracji. Aby umożliwić działanie automatu zarówno na platformie Windows jak i Linux został on napisany w języku Python. Wymagania dotyczące wiedzy użytkownika zależą w znacznej mierze od implementacji skryptu oraz specyfiki migrowanej usługi lub aplikacji.

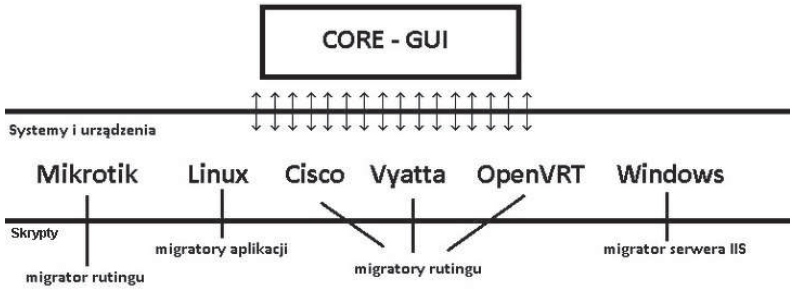
Niestety automat posiada pewne ograniczenia, bez których jego zrealizowanie byłoby znacznie trudniejsze lub wręcz nierealne. AM został zrealizowany z wykorzystaniem interpretera języka Python w wersji 2.7. Potrzebne też są do jego uruchomienia określone aplikacje (np. klient ssh/telnet, sshpass). Jeśli migracja przeprowadzana jest zdalnie, to komputery i urządzenia muszą być ze sobą połączone siecią LAN. Użytkownik powinien posiadać podstawową wiedzę w zakresie działania i konfiguracji usług oraz znać login, hasło oraz adres IP maszyny, na której ma być przeprowadzona migracja.

### 5.2 Sposób realizacji automatu

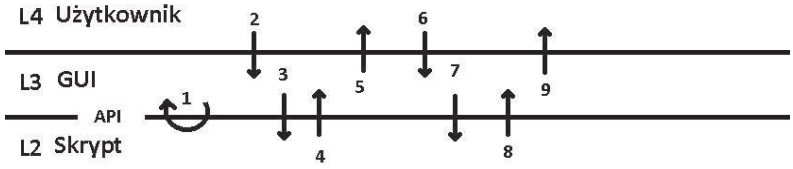
Aplikacja AM składa się z głównego elementu jakim jest „core" dalej występujący również pod pojęciem GUI ze względu na jego główna funkcję. Wszystkie migratory są traktowane jako moduły dla „core" (patrz rysunek 7) i zrealizowane w postaci skryptów. Wszystkie są podzielone za pomocą katalogów na systemy, dla których zostały napisane.

Architekturę aplikacji opisuje rysunek 8. Można ją podzielić na dwie podstawowe warstwy L2 oraz L3. Każda z tych warstw pełni inną funkcję w aplikacji. Aby można było zrealizować migrację należy uwzględnić dodatkową warstwę L4, która przedstawia użytkownika aplikacji.

Proces migracji składa się z 9 kroków. Pierwszym jest wczytanie z plików xml krótkich opisów migratorów oraz wyświetlenie ich na ekranie(1). Po tym kroku użytkownik podaje adres IP, hasło i login dla uprzywilejowanego użytkownika migrowanego systemu oraz wybiera interesujący go migrator (2). Na



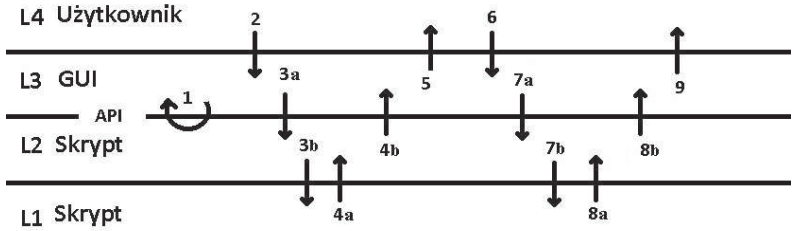

**Rysunek 7.** Logiczna struktura automatu




**Rysunek 8.** Struktura aplikacji wraz z procesem komunikacji z uwzględnieniem użytkownika

ekranie pojawi się informacja dokładniej opisująca wybrany migrator. Następnie aplikacja zapyta się skryptu o wymagane do migracji dane (3). Skrypt zwróci taką informację (4) i na ekranie użytkownika pojawia się formularz do uzupełnienia (5). Po wypełnieniu wszystkich pól przez użytkownika (6), GUI wpisuje parametry do pliku xml oraz wywołuje skrypt(7). Aplikacja nie posiada żadnej wiedzy w jaki sposób realizowane są zadania wewnątrz skryptu. Kiedy migrator zakończy swoje działanie, wysyła wynik do GUI (8), natomiast aplikacja wyświetla go dla użytkownika (9). Przed wyborem kolejnego skryptu GUI powinno przywrócić plik xml do stanu sprzed migracji.

W przypadku adaptacji gotowych migratorów pojawia się dodatkowa warstwa L1 (patrz rys. 9). Wynika to z konieczności adaptacji zrelizowanych już skryptów do współpracy z GUI. Funkcję komunikacji z aplikacją przejmuje dodatkowy skrypt, który pozwala na dostosowanie istniejących już skryptów do zdefiniowanego na jej potrzeby API. Kroki 3a, 3b oraz 4a, 4b odpowiadają za odpytanie skryptu o potrzebne parametry i odpowiednio przekazanie ich do GUI. W niektórych zastosowaniach kroki 3b oraz 4a będą opcjonalne, ponieważ liczba oraz typ potrzebnych parametrów będą zawsze takie same. Analogicznie rozszerzone zostały kroki 7 i 8. Ponieważ w tym modelu za migrację odpowiada warstwa L1 w kroku 7a i 7b przekazywane jest polecenie o rozpoczęciu migracji. Natomiast w krokach 8a i 8b odpowiednio skrypt warstwy L1 przekazuje wynik migracji do warstwy L2 oraz warstwa L2 wysyła otrzymany wynik do GUI.



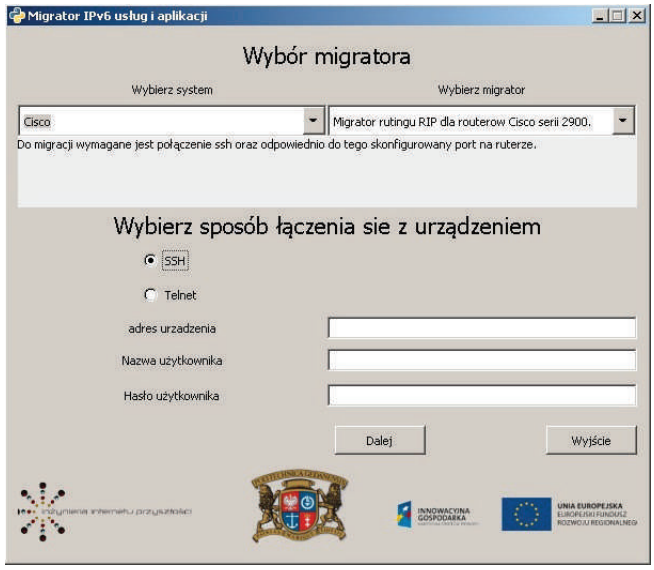

**Rysunek 9.** Struktura aplikacji wraz z procesem komunikacji z uwzględnieniem użytkownika

### 5.3 Wygląd oraz obsługa aplikacji

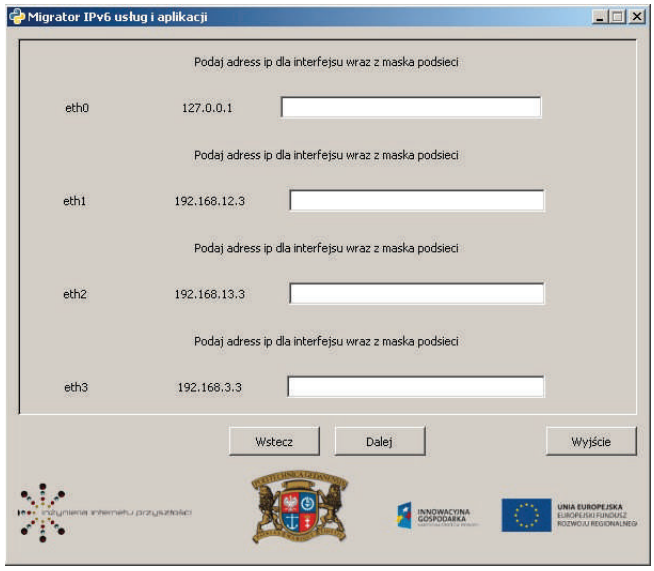
Interfejs automatu AM zarówno dla systemu Windows jak i Linux wygląda niemal identycznie. Interfejs użytkownika przedstawiono na rysunkach 10 oraz 11. Pierwszy z nich przedstawia ekran, na którym wybieramy migrator oraz wprowadzamy dane migrowanej maszyny. Drugie prezentuje zaś niezbędne do migracji parametry.

Z punktu widzenia użytkownika obsługa automatu AM wygląda następująco:

1. Użytkownik wprowadza w odpowiednie pola adres IPv4 maszyny, na której będzie przeprowadzana migracja oraz login i hasło uprzywilejowanego użytkownika systemu (rys. 10).
2. Z wyświetlonej przez program listy migratorów użytkownik wybiera interesujący go migrator i po przeczytaniu opisu decyduje się na migrację klikając przycisk „Dalej” (rys 10).
3. Użytkownik uzupełnia pola, które aplikacja wyświetla na ekranie. Są to parametry niezbędne do przeprowadzenia migracji (rys 11).
4. Użytkownik potwierdza wprowadzone dane klikając OK.
5. Po dokonaniu migracji użytkownik może przeczytać wynik działania migratora na ekranie.



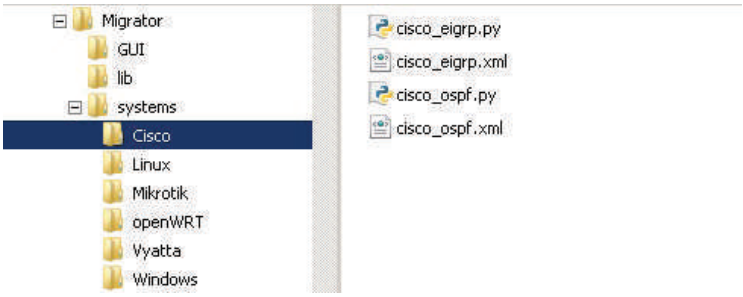

**Rysunek 10.** Struktura aplikacji wraz z procesem komunikacji z uwzględnieniem użytkownika




**Rysunek 11.** Struktura aplikacji wraz z procesem komunikacji z uwzględnieniem użytkownika

### 5.4 Implementacja aplikacji

Na potrzeby projektu przyjęta została prosta konwencja nazewnictwa oraz struktura katalogów. Nazwy folderów są równoznaczne z nazwami systemów. Wszystkie foldery systemowe znajdują się w folderze "Systems", pliki należące do "core" w folderze „GUI", natomiast biblioteki, z których korzysta aplikacja lub skrypty w folderze "lib". Strukturę tą przedstawia rysunek 12.




**Rysunek 12.** Struktura katalogów automatu migracji

Przyjęte nazwy plików:
Skrypty poziomu L2: nazwaSystemu_funkcjonalnosc np. cisco_eigrp.
Pliki xml: identyczna jak nazwa skryptu L2 z rozszerzeniem *.xml.
Skrpty poziomu L1: z racji opcjonalności tej warstwy, skrypty mogą posiadać dowolną nazwę, jednak nazwa związana z funkcjonalnością będzie dobra praktyką, pozwalającą zachować pewien porządek.

Struktura pliku xml wygląda następująco:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE system SYSTEM "information.dtd">
<system>
      <information>
            <connection></connection>
            <ip></ip>
            <login></login>
            <password></password>
      </information>
      <migrator>
            <description></description>
            <explanation></explanation>
      </migrator>
      <parameters>
            <parameter>
                  <id></id>
                  <question></question>
                  <name></name>
                  <cfg_value></cfg_value>
                  <type></type>
                  <value></value>
                  <result></result>
      </parameter>
```

```
    </parameters>
    <result_migration></result_migration>
</system>
```

Tag <system> jest korzeniem drzewa. Tagi <information> oraz <migrator> służą odpowiednio do podania informacji dotyczących logowania się do migrowanego systemu i opisu migratora. Opis pozostałych tagów w plikach xml definiuje tabela 2.

| **Tag** | **Typ** | **Opis** |
|---|---|---|
| <connection> | ciąg znaków | typ połączenia ze zdalną maszyną, możliwe wartości to telnet lub ssh |
| <ip> | IPv4, IPv6 | adres IPv4 lub IPv6 |
| <login> | dowolny ciąg znaków | login dla uprzywilejowanego użytkownika |
| <password> | dowolny ciąg znaków | hasło dla uprzywilejowanego użytkownika |
| <description> | dowolny ciąg znaków | krótki opis migratora, nazwa oraz funkcje |
| <explanation> | dowolny ciąg znaków | dłuższy opis migratora, np. co i w jaki sposób realizuje lub co jest potrzebne do jego uruchomienia |
| <id> | dowolne unikatowe | unikatowy identyfikator parametru |
| <question> | dowolny ciąg znaków | opis do wyświetlenia w formularzu |
| <name> | dowolny ciąg znaków | nazwa związana z migrowanym parametrem np. eth0 |
| <type> | number, string, radio, IPv4 lub IPv6 | określa jakiego typu powinno być pole <value> |
| <cfg_value> | number, string, radio, IPv4 lub IPv6 | wartość jaka była obecna w konfiguracji |
| <value> | number, string, IPv4 lub IPv6 | wartość wymagana do przeprowadzenia migracji |
| <result> | ciąg znaków | zwraca informację na temat powodzenia migracji danego parametru, stworzone z myślą o przyszłym, dokładnym testowaniu aplikacji |
| <result_migration> | ciąg znaków | zwraca informację do GUI czy poprawnie wykonała się wywoływana funkcja, przyjmuje wartość 0 jeśli prawidłowo się wykonała |

**Tablica 2.** Opis tagów w plikach xml.

Definicje poszczególnych typów: *number* – oznacza liczbę; string – oznacza ciąg znaków; *radio* – oznacza, że parametr ma być wyświetlony w postaci „radio przycisku"; *IPv4* – oznacza adres IPv4; *IPv6* – oznacza adres IPv6 wraz z maską w postaci „/liczba".

Aby zrealizować proces komunikacji z uwzględnieniem użytkownika przedstawiony na rysunku 8, zostało zdefiniowane uniwersalne API. Zadanie komunikacji jest realizowane za pomocą dwóch funkcji. Każdy skrypt warstwy L2 powinien posiadać zaimplementowaną ich obsługę, aby mógł współpracować z aplikacją. GUI wywołuje skrypty z parametrem, którym jest nazwa funkcji. Pierwszą z nich jest funkcja *needed_params*. Służy ona do komunikacji ze skryptem (rys. 8 - krok 3) w celu zapytania o wymagane do migracji dane. Skrypt po wywołaniu go z takim parametrem dodaje do pliku xml, do gałęzi <parameters>, pożądane dane. Funkcją pobierająca wpisane przez użytkownika nowe wartości wymaganych parametrów oraz wykonująca właściwą migrację jest *migrate*. Obie funkcje powinny wpisywać wartość 0 lub opis wykonanych czynności do tagu <result_migration>. GUI interpretuje to jako poprawne zakończenie się funkcji. W przypadku wartości różnej od zera użytkownik jest informowany o niepoprawnym przebiegu migracji. Jeśli w tym polu pojawi się ciąg znaków dłuższy niż trzy, to zostanie on wyświetlony jako komunikat na ekranie użytkownika.

## 6 Podsumowanie

Biorąc pod uwagę nadal niewielki udział sieci działających w oparciu o IPv6 oraz praktyczny brak konkurencji w przypadku gotowych rozwiązań systemów migracji aplikacji z IPv4 do IPv6, przydatność zrealizowanego systemu automatycznej migracji może okazać się znacząca. Mniej doświadczeni administratorzy zazwyczaj wolą przeprowadzać procedurę migracji krok po kroku konfigurując aplikację. Bardziej doświadczeni zazwyczaj tworzą pewne skrypty usprawniające proces migracji w mniejszym lub większym stopniu. Jedni i drudzy mogliby z powodzeniem posłużyć się odpowiednim gotowym rozwiązaniem, które zapewniałoby elastyczność (np. migracja różnych aplikacji na różne sposoby). Zarówno tym pierwszym jak i tym drugim zależy przecież na szybkich, ale i bezproblemowych oraz poprawnych wdrożeniach.

Obecnie nie ma na rynku gotowych narzędzi czy projektów zajmujących się tym zagadnieniem. Warto jednak podkreślić, że realizowany w ramach IIP Przewodnik migracji można potraktować jako ważny krok, do przeprowadzenia kompleksowej migracji środowiska sieciowego IPv4 do IPv6. Przewodnik migracji nie tylko rozpoznaje urządzenia sieciowe, systemy operacyjne, usługi i aplikacje IPv4 i wskazuje na zakres możliwości ich migracji do IPv6 oraz udziela porad konfiguracyjnych dotyczących poszczególnych elementów sieci, ale również umożliwia testowanie poprawności przeprowadzenia migracji do IPv6. Jest również źródłem wiedzy i dobrych praktyk wdrażania IPv6.

Kolejnym krokiem w celu ułatwienia procesu migracji jest tworzenie aplikacji, które w pewien sposób go ułatwią lub wręcz zautomatyzują. W ramach IIP zrealizowano klika aplikacji, które zajęły się tą tematyką. Automaty migracji są w stanie dokonywać zarówno migracji aplikacji oraz usług. Powstała również aplikacja, która w pewnym stopniu ujednolica powstałe migratory oraz upraszcza ich obsługę. Fakt ten dowodzi bez wątpienia, że automatyzacja migracji jest

możliwa do zrealizowania. Należy jednak pamiętać o ograniczeniach z nią związanych.

## Literatura


1. K.Nowicki, K. Podolski, W. Gumiński, K. Wasilewski, Ł.Łoska, P. Kamiński: *Automatyzacja procesu migracji do protokołu IPv6* // Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne. - 2012, nr 8-9, s. 1332-1341 - ISSN 1230-3496
2. N. R. Murphy, D. Malone, *IPv6 Network Administration*, O'Reilly Media 2005
3. Internet Society, World IPv6 Day, The official website `http://www.worldipv6day.org/`
4. Światowy Dzień IPv6 na Politechnice Gdańskiej, The official website `http://kti.net.pl/`
5. InformationWeek, IPv6 Makes Slow Progress, `http://www.informationweek.com/news/infrastructure/ipv6/212501014`, Maj 2011
6. W. Gumiński, J. Światowiak, *Obsługa protokołu IPv6 w systemach operacyjnych Windows i Linux*, Przegląd Telekomunikacyjny i wiadomości telekomunikacyjne, 8-9/2010, s. 797-798
7. K. Nowicki, J. Światowiak, B. Gajda, *Problemy wdrażania protokołu IPv6*, Przegląd Telekomunikacyjny, 8/9/2010
8. Inżynieria Internetu Przyszłości, The Official Website of Future Internet Engineering, `http://iip.net.pl`
9. W. Gumiński i in., *Przewodnik migracji do IPv6*, Przegląd telekomunikacyjny i wiadomości telekomunikacyjne, 8-9/2011, s. 1184-1192
10. K. Nowicki i in., *Knowledge Managemenent in the IPv6 Migration Process*, Computer Networks 2011, Czerwiec 2011, s. 114-124
11. K. Nowicki i in., *Extension management of a knowledge base migration process to IPv6*, The 11th IEEE IPSJ International Symposium on Applications and the Internet, Munich, July 2011, s. 497-501
12. K. Nowicki i in., *Przewodnik migracji IPv6 - narzędzie dla administratora sieci SOHO*, // W : Inżynieria internetu przyszłości - Część 1 / praca zbiorowa pod red. W. Burakowski, P. Krawiec. - Warszawa : Oficyna Wydaw. PW, 2012. - S. 176-184 : ISBN 978-83-7814-042-9
13. `http://www.nist.gov/el/msid/expect.cfm`
14. Microsoft TechNet, *Teredo tunneling overview*, `http://technet.microsoft.com`
15. S. Deering, R. Hinden, *Internet Protocol, Version 6 (IPv6) Specification*, RFC 2460, IETF, 1998
16. R. Hinden, S. Deering, *IP Version 6 Addressing Architecture*, RFC 4291, IETF, 2006
17. Cisco IOS Documenation, `http://www.cisco.com`
18. Vyatta Documentation, `http://www.vyatta.com`
19. OpenWRT Web Site, `http://www.openwrt.org`
20. K. Podolski, K.Nowicki: *The Ipv4 To Ipv6 Migration Of Applications*, ICT Young 2012, Gdańsk, 2012
21. Apache Project Web Site, `http://www.apache.org`
22. The ProFTPD Project Web Site, `http://www.proftpd.org`
23. M. Garski, S. Gorczyczewski, B. Grabowski: *Automatyczny system instalacji sieci IPv6 dla firm i użytkowników indywidualnych*, Projekt inż., WETI Politechnika Gdańska 2010

# Protokoły SNMP oraz IPsec w sieciach wykorzystujących IPv6

Bartosz Gajda[1], Wiktor Procyk[1], Piotr Skałecki[1], Jan Kwiatkowski[2], Krzysztof Chudzik[2], Kamil Nowak[2]

[1] Poznańskie Centrum Superkomputerowo Sieciowe
[2] Instytut Informatyki, Politechnika Wrocławska

**Streszczenie** Rozwój sieci i skuteczne zarządzanie jej zasobami wymaga odpowiednich narzędzi do monitorowania, kontroli oraz rozliczeń z użytkownikami. Ponadto w sieciach VPN dane przechodzą przeważnie przez sieć nad którą nie mamy kontroli. Wymaga to wprowadzenia mechanizmów zabezpieczających dane podczas przesyłania. W niniejszej pracy przedstawiono problemy zarządzania oraz aspekty zapewnienia bezpieczeństwa informacji w sieci IPv6. Przedstawiono koncepcję wykorzystania aplikacji *pmacct − snmp* do zarządzania sieciami IPv6 oraz wykorzystanie zbioru protokołów IPsec w celu zapewnienia bezpieczeństwa przesyłanych danych.

## 1 Wprowadzenie

Rozwój sieci i skuteczne zarządzanie jej zasobami wymaga odpowiednich narzędzi do monitorowania, kontroli oraz rozliczeń z użytkownikami. Narzędzia te dając bezpośredni podgląd na aktualny stan sieci i umożliwiając szybką reakcję na konkretne zdarzenia czy negatywne prognozy, pozwalają unikać sytuacji niekorzystnych dla użytkowników. Wprowadzenie protokołu IPv6 z pewnością zwiększy liczbę urządzeń podłączonych do sieci i tym samym wymagających zarządzania. Ciągłej obserwacji i kontroli stanu ich działania będzie wymagać coraz większa ilość aplikacji i usług wykorzystujących coraz to bardziej wyszukane rozwiązania i technologie. Niestety większość istniejących aplikacji zarządzających oraz sprzętu sieciowego nie potrafi używać protokołu SNMP w środowisku IPv6. Pojawia się przykładowo problem w zarządzaniu środowiskami hybrydowymi łączącymi sieci IPv4 i IPv6. Jest to z pewnością jedną z przyczyn opóźniających wdrożenie protokołu IPv6 w firmach, instytucjach i przede wszystkim w sieciach operatorskich.

Z drugiej strony rozwojowi sieci IP towarzyszy nieustanny wzrost liczby sieci wirtualnych VPN (Virtual Private Networks), które służą do logicznych połączeń pomiędzy rozproszonymi zasobami firm i instytucji. Rozwój sieci VPN jest również nierozerwalnie związany z rozwojem wirtualizacji zasobów sieciowych oraz obliczeniowych, gdzie każdy wirtualny system generuje potrzebę wytworzenia sieci wirtualnej łączącej go ze światem zewnętrznym. Sieci VPN umożliwiają tworzenie sieci prywatnych bez tworzenia własnej infrastruktury sieciowej.

W tym przypadku jako medium pośredniczące w połączeniach jest wykorzystywana przede wszystkim globalnie dostępna sieć Internet. Firma tworząca połączenie VPN nie ponosi kosztów budowy własnej infrastruktury bądź dzierżawy łączy. Sieci wirtualne umożliwiają również optymalizację wykorzystania zasobów sieciowych. Jednym połączeniem fizycznym może być połączonych wiele sieci logicznych. Ta właściwość sieci VPN jest wykorzystywana przede wszystkim przez dużych operatorów sieciowych oraz operatorów chmur obliczeniowych.

Naturalnymi wadami sieci typu VPN jest trudność zapewnienia w nich jakości usług oraz bezpieczeństwa informacji. Dane przechodzą przeważnie przez sieć, nad którą nie mamy kontroli. Technologią stworzoną na potrzeby wirtualizacji sieci oraz równocześnie zapewniającą bezpieczeństwo przesyłanych danych jest zbiór protokołów określany skrótem IPsec (Internet Protocol Security). Technologia ta nie jest zwartym standardem, ale zbiorem zaleceń, które dwie strony połączenia powinny stosować, aby zapewnić sobie bezpieczne połączenie wirtualne. W praktyce w miarę prosty sposób możliwe jest połączenie urządzeń wykorzystujących oprogramowanie jednego producenta. Niestety połączenie ze sobą urządzeń z odmiennym oprogramowaniem jest bardzo trudne lub wręcz niemożliwe. Co więcej różne implementacje zbioru protokołów IPsec mogą wpływać na parametry utworzonych przez nie sieci VPN (opóźnienia, obciążenie procesora, przepustowość sieci, itp.). Połączenie IPsec może być ustanowione w trybie transportowym lub tunelowym. W pierwszym przypadku przetwarzana jest jedynie zawartość pakietu IP (dane). W drugim trybie przetwarzany jest cały pakiet wraz z nagłówkiem IP. Następnie do tak przetworzonego pakietu dodawany jest nowy nagłówek IP. Ten tryb jest kluczową własnością technologii IPSec, która daje możliwość tworzenia sieci VPN.

W niniejszej pracy przedstawiono problemy zarządzania oraz aspekty zapewnienia bezpieczeństwa informacji w sieci IPv6. W sekcji drugiej przedstawiono koncepcję wykorzystania aplikacji *pmacct – snmp* do zarządzania sieciami IPv6 przez aplikacje przystosowane do pracy w sieci IPv4. Sekcje 3 oraz 4 poświęcone są zagadnieniom wykorzystania protokołu IPsec w sieci IPv6. Wyniki badań narzutu wnoszonego przez protokół IPsec na działanie aplikacji przedstawione są w sekcji 5. Sekcja 6 zawiera podsumowanie przeprowadzonych prac.

## 2 Wykorzystanie protokołu SNMP w sieciach IPv6

Dostępne obecnie na rynku aplikacje do zarządzania sieciami nie zapewniają wystarczającego wsparcia dla protokołu IPv6, co więcej wiele z nich nie wspiera go wcale. Większość z tych aplikacji wykorzystuje jednak najpopularniejszy obecnie protokół zarządzania, jakim jest SNMP [1] (Simple Network Management Protocol), co daje im dostęp do wszelkich informacji udostępnianych przez urządzenia sieciowe oraz kontrolę stanu ich działania. Użycie tego protokołu do zdobywania przez stację zarządzającą odpowiednio wcześniej zagregowanych i przefiltrowanych danych jest metodą, pozwalającą na pomiary oraz wizualizację dowolnego ruchu w sieci, poddanego obserwacji przez wykorzystaną w tym celu aplikację pośrednią. Agregacja i filtrowanie ruchu mają również duże znaczenie ze wzglę-

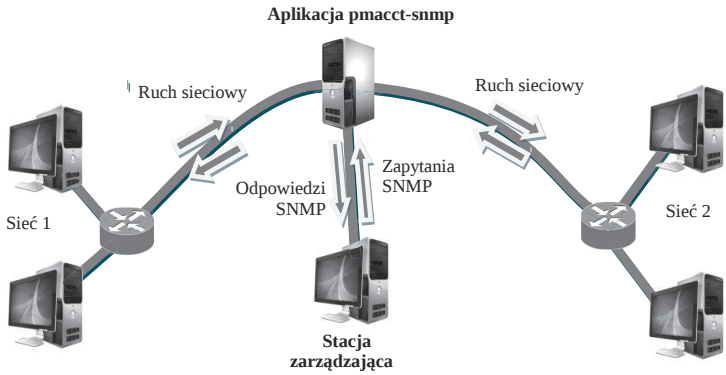
dów wydajnościowych, ponieważ obszerne sieci potrafią w bardzo krótkim czasie wytworzyć znaczne ilości danych, co tym bardziej odczuwalne jest w sieciach IPv6. Kluczem do szeregu operacji składających się na zarządzanie jest uzyskanie odpowiedniej szczegółowości informacji, przede wszystkim natomiast dotyczy to rozliczeń adresów IP, a przez to wyszczególnienie konkretnych użytkowników i urządzeń, również tych nie udostępniających agenta SNMP. Możliwości takie oferuje aplikacja *pmacct* [2], która dzięki umiejętności gromadzenia danych z wielu różnych źródeł jednocześnie (libpcap, Netlink/ULOG, NetFlow, sFlow oraz IPFIX), pozwala je w prosty sposób przetwarzać, by ostatecznie w odpowiedniej postaci przekazać dalej do stacji zarządzającej przez jeden z wielu protokołów zarządzania lub umieścić w bazie danych SQL. *Pmacct* pozwala klasyfikować i agregować zgromadzone dane na wiele różnych sposobów, według takich parametrów jak: VLAN, źródłowy i docelowy adres MAC, host, port, rodzaj protokołu IP, pole ToS/DSCP.

Udostępnienie przygotowanej wcześniej porcji informacji przez protokół SNMP, wymaga użycia rozszerzenia dla aplikacji *pmacct*, które współpracując z agentem SNMP, pozwoli stacji zarządzającej na uzyskanie dostępu do informacji o ruchu w sieci. Zadanie to dotychczas dla sieci IPv4 doskonale spełniał $pmacct - snmp$, będący nakładką na agenta Net-SNMP oraz na samego *pmaccta*. Współpracując z obydwoma aplikacjami, pozwala *pmacctowi* udostępniać przygotowane dane za pomocą SNMP do dalszego wykorzystania przez stację zarządzającą. Poznańskie Centrum Superkomputerowo-Sieciowe w ramach projektu Inżynieria Internetu Przyszłości rozwija aplikację $pmacct - snmp$, co pozwala na wykorzystanie obecnie istniejących narzędzi do monitorowania ruchu w sieci IPv4 przy użyciu protokołu SNMP, również do monitorowania ruchu w sieci IPv6.

Na rysunku 1 przedstawiono sposób, w jaki odbywa się pasywne zbieranie informacji dotyczących ruchu sieciowego, przetwarzanie ich i udostępnianie poprzez protokół SNMP stacji zarządzającej. Funkcję tę pełni host z zainstalowaną aplikacją $pmacct - snmp$. Za wykorzystaniem aplikacji *pmacct* przemawia również jej wszechstronność i możliwość uruchomienia na różnych systemach operacyjnych takich jak Linux, BSD, Solaris, czy systemy wbudowane, co pozwala na pracę jej nawet na niewielkich domowych routerach.

Najprostszym sposobem rozwiązania problemu utrudnionego zarządzania sieciami IPv6 jest użycie istniejących narzędzi do zarządzania w sieci IPv4 i przystosowanie ich do pracy w sieciach IPv6. Jako że protokół SNMP z powodzeniem wykorzystywany jest od wielu lat w większości sieci i tak samo obsługiwany jest przez większość programów do zarządzania, wykorzystany został jako medium do wymiany informacji pomiędzy urządzeniami monitorowanymi, a urządzeniem zarządzającym. Gromadzenie i udostępnianie przez SNMP danych na temat urządzeń znajdujących się w sieci oraz ruchu pomiędzy nimi odbywa się następująco:

- aplikacja *pmacct* gromadzi, agreguje i filtruje dane pochodzące z wybranego źródła i przekazuje je do kolejki komunikatów (ang. „named pipe"),
- agent SNMP poprzez nakładkę $pmacct - snmp$ odczytuje zebrane informacje z kolejki komunikatów i udostępnia je w sieci.




**Rysunek 1.** Gromadzenie informacji przez aplikację *pmacct – snmp* i udostępnianie ich stacji zarządzającej.

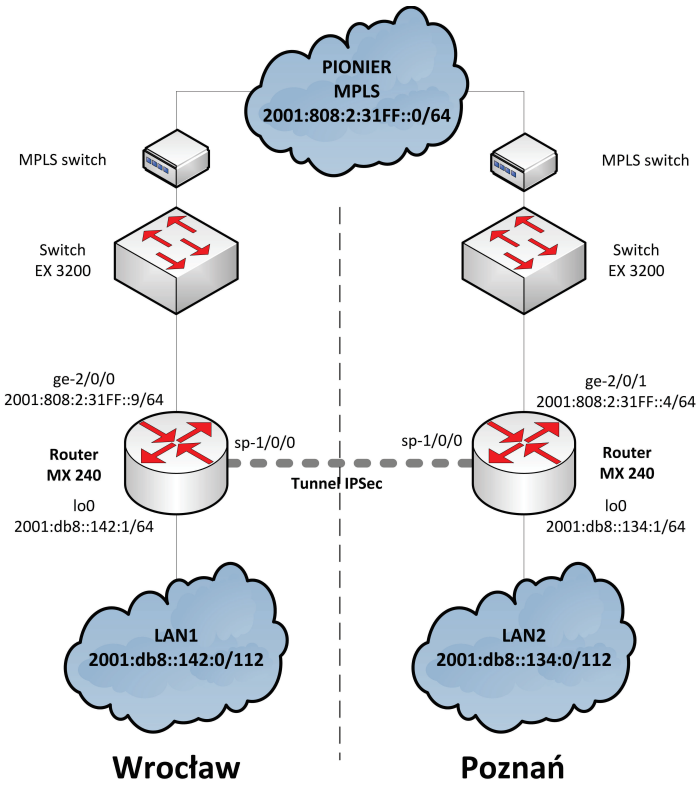
Dotychczasowe działanie *pmacct – snmp* opierało się na tworzeniu tablicy w pamięci dla wszystkich zadeklarowanych przez użytkownika adresów sieciowych, co w przypadku sieci IPv4 było podejściem skutecznym ze względu na stosunkową małą pulę adresową. Dla sieci IPv6 jest to rozwiązanie bardzo nieefektywne. Na testowej maszynie wirtualnej z 1 GB pamięci RAM, wirtualnym procesorem i5 M450 i zainstalowanym systemem Ubuntu 11.04 już przy kilku tysiącach zadeklarowanych adresów (przy czym dla każdego z nich rejestrowanych jest kilka parametrów), potrafi dojść do sytuacji, że agent SNMP przestaje odpowiadać. Problem ten został rozwiązany przez dynamiczne generowanie listy adresów, dla których tworzone są statystyki i przekazywanie tej listy do agenta SNMP. Dzięki temu eliminowane są niepotrzebne dane. Zbierane są jedynie istotne informacje dotyczące aktualnego ruchu w wybranych do monitorowania sieciach.

Z uwagi na szesnastkową postać adresów IPv6 i ograniczenie w bazie MIB (Management Information Base) jedynie do zapisu nazw obiektów w postaci dziesiętnej, ustalony został sposób kodowania adresów, tak by w prosty i czytelny sposób można było poruszać się po bazie oraz aby struktura adresów została zachowana. W celu ujednolicenia zapisu, każdy z adresów zostaje w pierwszej fazie rozwinięty do maksymalnej postaci, a następnie każdy z 16-bitowych bloków zostaje przekształcony na postać dziesiętną, tworząc w ten sposób nową gałąź w drzewie MIB. Na końcu każdego z adresów dodano osobny indeks reprezentujący dowolną, ustaloną wcześniej wartość podlegającą badaniu, natomiast pierwszy z indeksów o numerze „1” zawiera słowną wartość, reprezentującą dany adres IPv6 w postaci szesnastkowej.

Rozwiązanie takie jest bardzo intuicyjne i pozwala w prosty sposób zorientować się administratorowi sieci, którego hosta dotyczy dany wpis. Poza tym te same adresy IP mają w drzewie obiektów przydzieloną zawsze tę samą nazwę, dzięki czemu łatwo je odszukać w bazie MIB.

```
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1838.1 = STRING: "fe80:0000:0000:0000:0a00:27ff:fee4:072e"
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1838.2 = INTEGER: 263651848
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1838.3 = INTEGER: 5
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1838.4 = INTEGER: 186278
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1838.5 = INTEGER: 0
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1839.1 = STRING: "fe80:0000:0000:0000:0a00:27ff:fee4:072f"
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1839.2 = INTEGER: 263649168
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1839.3 = INTEGER: 5
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1839.4 = INTEGER: 186279
iso.3.6.1.4.1.21695.1.4.2.6.65152.0.0.0.2560.10239.65252.1839.5 = INTEGER: 0
```

**Rysunek 2.** Przykład zapisu informacji o ruchu przychodzącym i wychodzącym w drzewie MIB




**Rysunek 3.** Topologia fizyczna sieci IPsec pomiędzy PWr i PCSS.

## 3 Protokół IPsec w środowisku sprzętowym wykorzystującym IPv6

Poniżej przedstawiono konfigurację połączenia IPsec w rozbudowanym środowisku sieciowym zawierającym dedykowane urządzenia sprzętowe.

Jako hosty końcowe posłużyły routery Juniper MX240 z zainstalowanymi kartami Multiservices DPC - sprzętowego wsparcia dla IPsec. Na rysunku 3 została przedstawiona topologia fizyczna sieci. Routery MX240 są rozlokowane

w laboratoriach PWr (Politechnika Wrocławska) i PCSS (Poznańskie Centrum Sieciowo-Superkomputerowe) połączonych ze sobą w technologii MPLS poprzez sieć Pionier. Do sieci Pionier routery zostały podłączone poprzez przełączniki Juniper EX3200. Węzłom końcowym połączenia IPsec (routery) zostały przydzielone adresy IPv6 z sieci 2001:808:2:31FF::0/64. Dodatkowo na interfejsach loopback routerów zostały zdefiniowane sieci wewnętrzne LAN1 i LAN2 o adresach 2001:db8::142:0/112 (PWr) i 2001:db8::134.0/112 (PCSS).

Połączenie IPsec zostało skonfigurowane w trybie tunelowym $Site-to-Site$ pomiędzy wirtualnymi interfejsami karty DPC (sp-1/0/0). Należy wspomnieć, że na routerach wyposażonych w karty sprzętowego wsparcia usług konfigurację IPsec przeprowadza się w nieco odmienny sposób niż na urządzeniach nie posiadających tych kart. Na routerach Juniper serii MX użycie karty DPC jest niezbędne do konfiguracji IPsec. Połączenie IPsec, jak tego należało się spodziewać, przebiegło w dwóch etapach:

1. Ustanowienie bezpiecznego kanału połączeniowego SA (Security Association) zostało przeprowadzone za pomocą protokołu IKE (ISAKMP).
2. Transfer zaszyfrowanych informacji pomiędzy PWr (sieć 2001:db8::142:0/112) i PCSS (sieć 2001:db8::134.0/112) został zrealizowany poprzez protokół ESP.

W celu weryfikacji i wizualizacji przebiegu negocjacji IKE, a następnie transferu zaszyfrowanych informacji został wykorzystany sniffer *Wireshark* uruchomiony na komputerze z systemem Linux. Komputer ze sniferem został podłączony do portu monitorującego skonfigurowanego specjalnie w tym celu na przełączniku Juniper EX 3200. Rysunek 4 przedstawia pakiety wymieniane pomiędzy węzłami końcowymi IPsec w fazie negocjacji IKE.

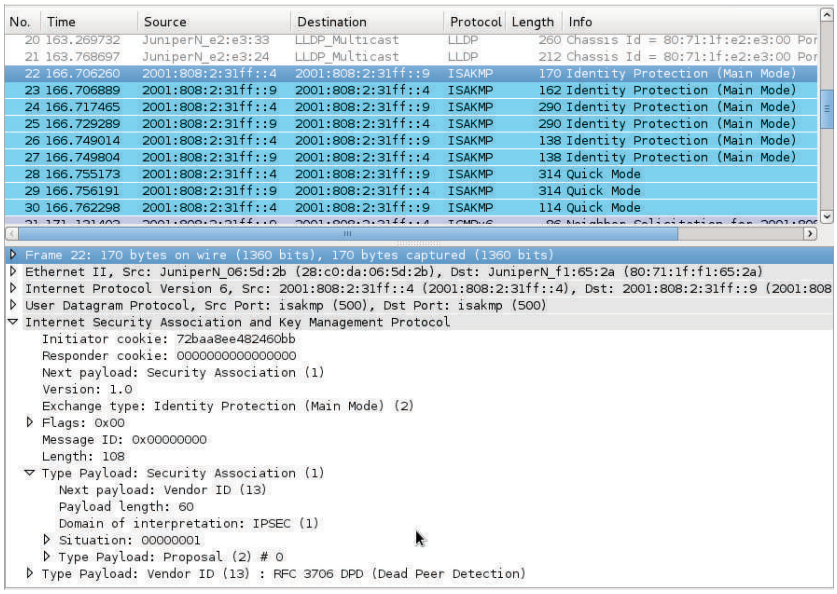
Po zakończeniu fazy 1 na routerach zostały automatycznie stworzone bezpieczne kanały komunikacyjne SA, co potwierdziło poprawność negocjacji IKE.

Następnie wykonano prosty test mający na celu wymuszenie bezpiecznej komunikacji IPsec pomiędzy sieciami LAN1 i LAN2. W tym celu użyto komendy ping ze zdefiniowanym adresem źródłowym z sieci LAN1 i docelowym z sieci LAN2. Uzyskano poprawną odpowiedź ze zdalnego hosta, co potwierdziło poprawność połączenia IPsec. Ruch sieciowy, podobnie jak poprzednio, został zarejestrowany programem *Wireshark*. Wyniki testu przedstawia rysunek 5.

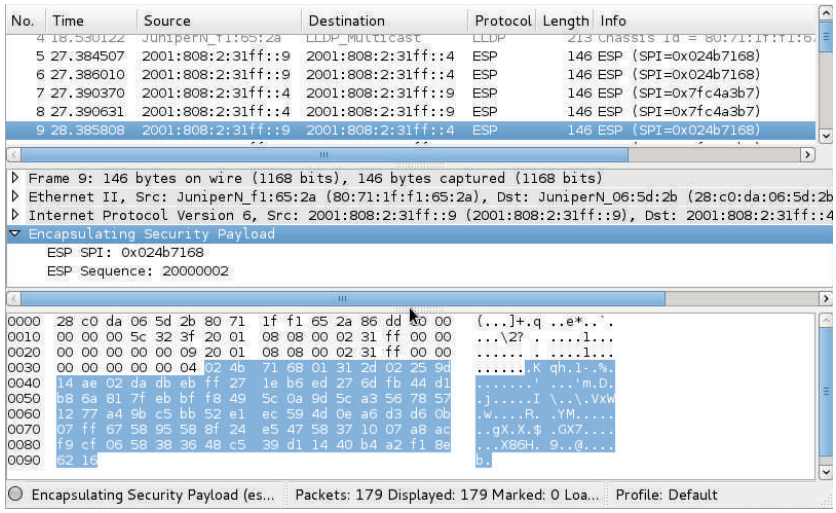
Przeprowadzone doświadczenie potwierdziło poprawność konfiguracji i działania połączenia IPsec w rozbudowanym środowisku sieciowym ze sprzętowym wsparciem dla tego protokołu.

## 4 Protokół IPsec w środowiskach wirtualizacyjnych wykorzystujących IPv6

*Moodle*, jako system zdalnego nauczania, jest powszechnie stosowany na uczelniach polskich jak i zagranicznych. Jest on uruchamiany jako aplikacja www i praktycznie stanowi zbiór skryptów PHP. Klient, chcący uzyskać dostęp do aplikacji, wykorzystuje przeglądarkę internetową. Aplikacja ta stosowana jest

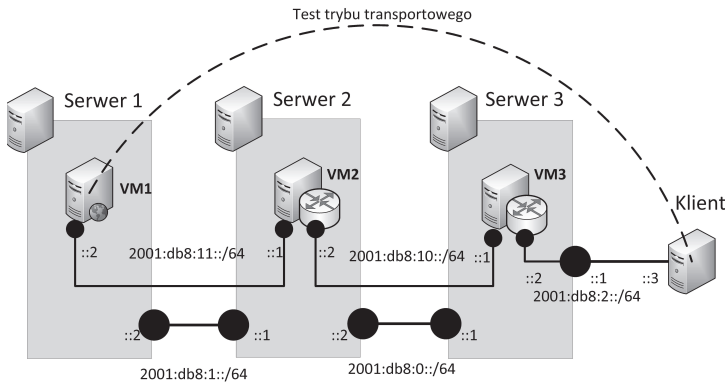



**Rysunek 4.** *Wireshark* - faza 1 połączenia IPsec, negocjacja IKE (ISAKMP).




**Rysunek 5.** *Wireshark* - Faza 2 połączenia IPsec, wymiana zaszyfrowanych danych protokołem ESP.

między innymi do przeprowadzania egzaminów co wymaga zapewnienia podczas




**Rysunek 6.** Środowisko pracy aplikacji *Moodle* w trybie transportowym IPsec.

jej pracy wysokiego poziomu bezpieczeństwa. W tym celu można zastosować technologię IPsec.

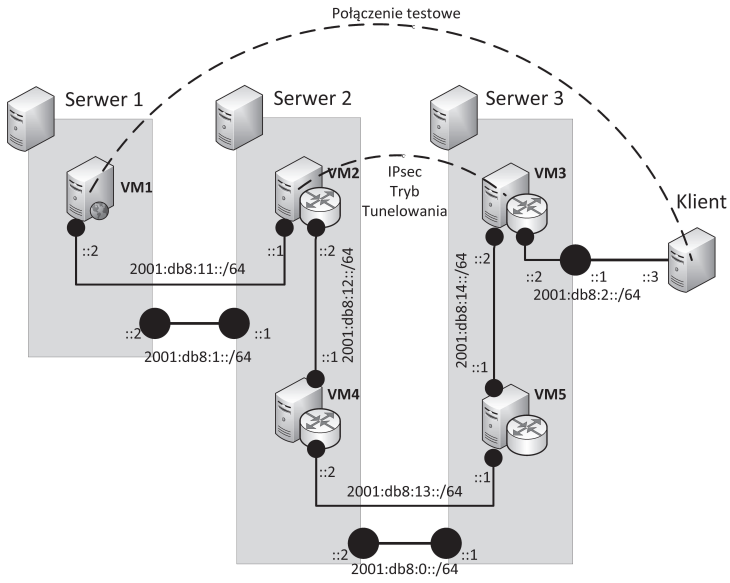
Na rysunku 6 przedstawiono testowane środowisko pracy aplikacji *Moodle* dla trybu transportowego IPsec. W prototypowej instalacji wybrano platformę Linux Debian 6 (Squeeze) [8], ze względu na jej popularność wśród administratorów systemów sieciowych. Na tej platformie uruchomiono środowisko wirtualizacyjne Xen w wersji 4. Maszyny wirtualne uruchamiane w ramach środowiska wirtualizacyjnego pracowały również pod kontrolą systemu Debian 6. Jako implementację IPsec wybrano pakiet StrongSwan [5], ze względu na jego wszechstronność.

Protokół IPsec skonfigurowano w trybie transportowym tak, aby zabezpieczał ruch pomiędzy komputerem *Klient* a maszyną wirtualną $VM1$. Klient korzystający z przeglądarki www na komputerze fizycznym *Klient*, nawiązywał za jej pomocą połączenie z aplikacją *Moodle* uruchomioną na maszynie wirtualnej $VM1$, wykorzystując protokół IPv6.

Zawartość pliku konfiguracyjnego *ipsec.conf* odpowiadającą za konfigurację połączenia transportowego IPsec na $VM1$ zamieszczono poniżej:

```
conn ipv6-tr \\
     left=2001:db8:11::2
     right=2001:db8:2::3 \\
     authby=secret \\
     esp=aes128-sha1 \\
     ike=aes128-sha-modp1024 \\
     type=transport \\
     compress=no \\
     auto=add \\
```

Aplikacja funkcjonowała poprawnie. Poprawność połączenia została zweryfikowana dodatkowo programem *Wireshark*. W kolejnym kroku zostało zbadane




**Rysunek 7.** Środowisko pracy aplikacji *Moodle* w trybie tunelowym IPsec.

połączenie IPsec w trybie tunelowym. Środowisko badawcze przedstawiono na rysunku 7.

Podobnie jak poprzednio, klient nawiązywał połączenie z komputera *Klient* z aplikacją *Moodle* uruchomioną na maszynie wirtualnej $VM1$. Protokół IPsec został skonfigurowany w trybie tunelowania pomiędzy maszynami wirtualnymi $VM2$ oraz $VM3$ tak, aby tunelowaniu poddawać ruch odbywający się pomiędzy sieciami brzegowymi 2001:db8:11::/64 oraz 2001:db8:2::64 (tunel typu $Site - to - Site$). Routery pośredniczące $VM4$ i $VM5$ nie były świadome istnienia sieci brzegowych, więc routing pomiędzy sieciami brzegowymi bez połączenia tunelowego nie był możliwy.

Zawartość pliku konfiguracyjnego *ipsec.conf* odpowiadającą za konfigurację połączenia tunelowego IPsec na $VM1$ zamieszczono poniżej:

```
conn ipv6-tun \\
     left=2001:db8:12::2 \\
     leftsubnet=2001:db8:11::/64 \\
     right=2001:db8:14::2 \\
     rightsubnet=2001:db8:2::/64 \\
     authby=secret \\
     esp=aes128-sha1 \\
     ike=aes128-sha-modp1024 \\
     type=tunnel \\
     compress=no \\
```

```
auto=add \\
```

Przeprowadzone testy wykazały, że aplikacja funkcjonowała poprawnie. Monitorowanie połączenia za pomocą programu *Wireshark* potwierdziło poprawność działania protokołu IPsec w środowisku zwirtualizowanym.

Powyższe badania pokazują, że IPsec może zostać użyty do zabezpieczania komunikacji klienta z aplikacją sieciową. W prezentowanym przykładzie wykorzystania aplikacji zdalnego nauczania *Moodle*, można wprowadzić dodatkowy poziom bezpieczeństwa, pozwalający szyfrować transmitowane dane i uwierzytelniać końce tunelu IPsec, niezależnie od mechanizmów samego systemu *Moodle*. IPsec oferuje bardzo zaawansowane mechanizmy szyfrowania i uwierzytelniania, które można dopasować do wyspecyfikowanych wymagań. Zabezpieczenie takie mogłoby być stosowane do zabezpieczenia komunikacji różnych komponentów tej aplikacji, nie tylko komunikacji z klientem, przykładowo, komunikacji aplikacji pracującej na serwerze www z bazą danych na zdalnym serwerze.

## 5 Narzut wnoszony przez protokół IPsec

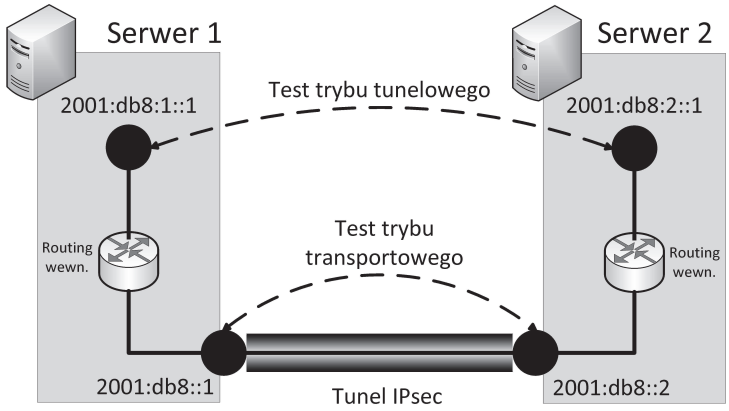
Celem badania było sprawdzenie poprawności funkcjonowania wybranych popularnych implementacji zestawu protokołów IPsec [4] w środowisku wykorzystującym protokół IPv6 oraz określenie narzutu, który wnoszony jest przez zbiór protokołów IPsec do transmisji danych. Testy przeprowadzono dla popularnych systemów operacyjnych Linux oraz MS Windows. Do testów wybrane zostały najnowsze dostępne wersje systemów. Testy platformy Linux przeprowadzono na dystrybucji Fedora 16. Testy platformy MS Windows przeprowadzono z wykorzystaniem MS Windows Server 2008 R2 Enterprise Edition.

W ramach testów przebadano następujące implementacje IPsec. Na platformie Linux testowano pakiety oprogramowania StrongSwan [5] oraz OpenSwan [6]. Na platformie MS Windows wykorzystano implementację IPsec wbudowaną w system. StrongSwan niedawno został włączony do dystrybucji Vyatta, która jest jedną z popularniejszych dystrybucji Linux wykorzystywanych do budowy infrastruktury sieciowej na bazie rozwiązań programowych.

Badaniom poddano oba tryby pracy protokołu IPsec: transportowy oraz tunelowy. Oceniano wpływ IPsec na następujące parametry, dostępne pasmo oraz opóźnienie w przesyle danych. Użytą metodą uwierzytelniania była metoda z kluczem współdzielonym (PSK - pre-shared key).

Środowisko badawcze składało się z dwóch komputerów fizycznych skonfigurowanych jak zilustrowano na rysunku 8. Komputery użyte do testów były komputerami klasy PC, z procesorem Pentium Dual-Core 3GHz, 4 GB RAM i kartami sieciowymi o przepustowości 1Gbps.

*Serwer*1 i *Serwer*2 były fizycznymi komputerami wyposażonymi w pojedyncze karty sieciowe skonfigurowane, odpowiednio, z adresami 2001:db8::1 oraz 2001:db8::2. Bezpośrednio pomiędzy tymi adresami uruchamiano protokół IPsec. Również bezpośrednio pomiędzy tymi adresami badano tryb transportowy IPsec. Testy trybu tunelowania IPsec wymagały konfiguracji dodatkowych interfejsów wewnętrznych. Skonfigurowano wirtualne interfejsy z adresami 2001:db8:1::1




**Rysunek 8.** Środowisko testowe dla badań protokołu IPsec współpracującego z protokołem IPv6.

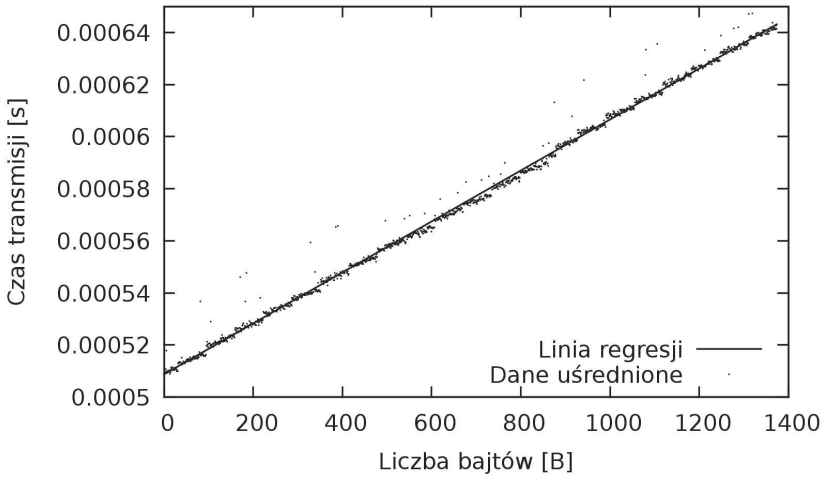
oraz 2001:db8:2::1. Używane były one jako źródło i cel ruchu sieciowego podlegającego tunelowaniu. Nie stosowano celowo żadnych dodatkowych zewnętrznych komputerów, czy innych źródeł ruchu sieciowego, aby zminimalizować ich wpływ na wyniki badań. Pakiety podlegały jedynie trasowaniu (routingowi) w obrębie stosu TCP/IP komputera fizycznego. Podkreślenia wymaga, że tunel protokołu IPsec zarówno w trybie transportowym jak i tunelowania zakończony był na interfejsach fizycznych z adresami 2001:db8::1 oraz 2001:db8::2.

Na platformie Linux zostały poprawnie skonfigurowane oba pakiety oprogramowania implementującego IPsec, to znaczy StrongSwan oraz OpenSwan zarówno w trybie transportowym jak i tunelowym. Konfiguracja była standardowa i nie odbiegała od sposobów konfiguracji podanych przez twórców oprogramowania [5,6]. W przypadku platformy MS Windows Serwer uruchomiono jedynie tryb transportowy.

Badanie dostępnego pasma polegało na zmierzeniu dostępnego pasma bez i z funkcjonującym IPsec. Do testów wykorzystano program *iperf* [7], który implementowany jest zarówno na platformie Linux jak i Windows. Rezultaty przeprowadzonych testów zebrano w poniższej tabeli.

| Platforma | Implementacja IPsec | Tryb transportowy | Tryb tunelowania |
|---|---|---|---|
| Linux | Bez IPsec | 891 Mbits/sec | 883 Mbits/sec |
| Linux | StrongSwan | 474 Mbits/sec | 460 Mbits/sec |
| Linux | OpenSwan | 467 Mbits/sec | 457 Mbits/sec |
| MS Windows | Bez IPsec | 795 Mbits/sec | — |
| MS Windows | Wew. systemu. oper. | 402 Mbits/sec | — |

Z przedstawionych wyników wynika, że włączenie protokołu IPsec powoduje zmniejszenie dostępnego pasma prawie o połowę na testowanych komputerach. W czasie badania zaobserwowano, że przesyłane pakiety są pofragmentowane.




**Rysunek 9.** Wyniki pomiarów czasu transmisji pakietów na platformie Linux z wykorzystaniem pakietu StrongSwan w trybie tunelowym

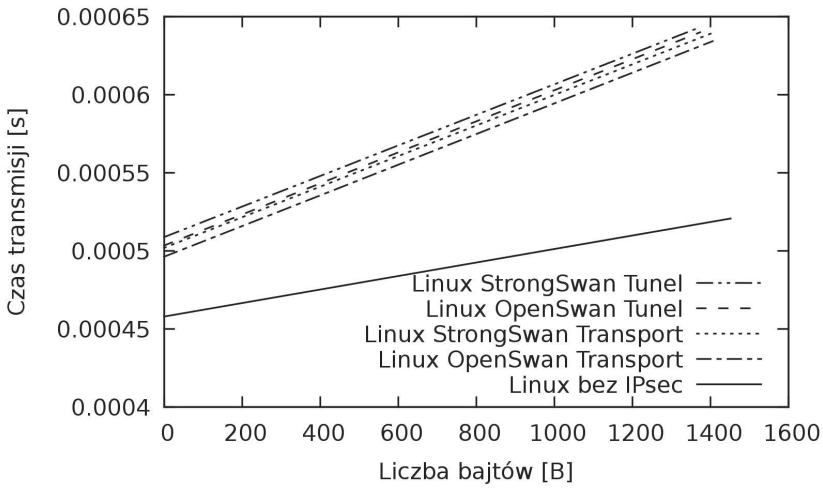
W kolejnych badaniach dotyczących narzutu wnoszonego przez IPsec zadbano aby nie dochodziło do fragmentacji.

Różnice pomiędzy systemami operacyjnymi czy implementacjami protokołu IPsec są nieznaczne, choć nieco lepszą wydajnością cechuje się w tym badaniu platforma Linux.

Celem kolejnego testu było zbadanie wpływu zastosowania grupy protokołów IPsec na opóźnienia w przesyle pakietów. Badanie polegało na wysłaniu pakietu danych z jednego komputera do drugiego. Po odebraniu, pakiet był natychmiast retransmitowany do komputera pierwszego. Do badań wykorzystano bezpołączeniowy protokół UDP. Na komputerze pierwszym mierzono okres czasu upływający od momentu rozpoczęcia wysyłania do momentu zakończenia odbioru danych. Program testujący działał w architekturze klient-serwer, gdzie klient generował dane, a serwer je odsyłał. Implementacji dokonano w języku Python.

Badania przeprowadzono dla różnych rozmiarów przesyłanych danych protokołu UDP. Rozmiar danych zmieniał się od zera do wartości maksymalnej, która dobrana była tak, aby nie następowała fragmentacja pakietów. Dla każdego rozmiaru danych wykonano po 1000 pomiarów, a następnie obliczono z nich średnią arytmetyczną. Przykładowe dane dla pomiaru czasu transmisji na platformie Linux z wykorzystaniem pakietu StrongSwan w trybie tunelowym przedstawiono na rysunku 9. Dla danych uśrednionych wyznaczono funkcję regresji liniowej. Wartości funkcji regresji są traktowane dalej jako wyniki do analizy i są one przedstawione na rysunkach 10, 11 dla platform Linux i Windows.

Wyniki pomiarów dla platformy Linux i Windows różnią się znacznie jeśli chodzi o wartości (wykresy mają różne zakresy wartości na osi Y i nie zaczynają



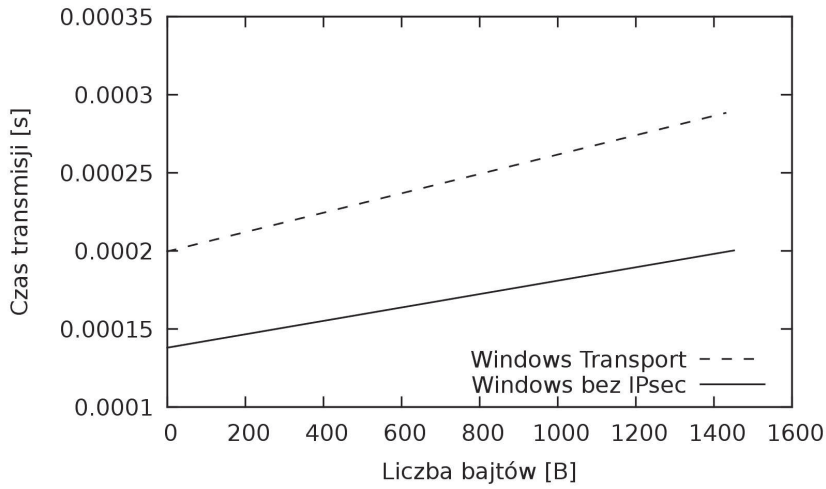

**Rysunek 10.** Wyniki pomiarów czasu transmisji pakietów na platformie Linux.

się w wartości 0 ze względu na czytelność wykresów). Może wynikać to z różnych implementacji stosu TCP/IP, sterowników, itp. Wymaga to osobnych badań wykraczających poza zakres tego opracowania. Niemniej jednak zauważyć można, że na obu platformach wpływ protokołu IPsec jest znaczący na opóźnienie transmisji pakietów. Badania dla platformy Linux pokazują, że wybór pakietu i trybu pracy IPsec ma niewielki wpływ na opóźnienie. Jak należało się spodziewać dla trybu transportowego jest ono mniejsze. Spośród badanych pakietów OpenSwan okazał się nieco szybszą implementacją.

## 6 Podsumowanie

Zaprezentowane rozwiązanie wykorzystujące aplikację *pmacct − snmp* pozwala bez większych nakładów w obecną infrastrukturę zarządzającą przejść z zarządzania siecią IPv4 do IPv6 dzięki wykorzystaniu obecnie działających narzędzi pracujących z SNMP lub poprzez zastosowanie innych, powszechnie używanych. Uniwersalność tej metody pozwala na wykorzystanie jej zarówno w małych sieciach domowych, jak i serwerowniach do śledzenia ruchu sieciowego i identyfikacji określonych zdarzeń. Sama aplikacja pmacct-snmp może wykorzystywać oprócz pmacct także inne agregatory przekazujące dowolne informacje dotyczące pracy poszczególnych maszyn i dzięki temu być uniwersalnym narzędziem usprawniającym nadzór nad zarządzanym środowiskiem.

Przeprowadzone badania zbioru protokołów IPsec wskazują na możliwość ich wykorzystania w celu zapewnienia bezpiecznej transmisji zarówno w sieciach rzeczywistych jak i wykorzystujących wirtualne zasoby sieciowe oraz obliczeniowe wykorzystujące protokół IPv6.




**Rysunek 11.** Wyniki pomiarów czasu transmisji pakietów na platformie Windows.

## Literatura

1. SNMP, http://www.snmp.com/
2. pmacct, http://www.pmacct.net/
3. Net-SNMP, http://www.net-snmp.org/
4. RFC4301:Security Architecture for the Internet Protocol, http://tools.ietf.org/html/rfc4301
5. Witryna projektu StrongSwan,http://www.strongswan.org
6. Witryna projektu OpengSwan, https://www.openswan.org
7. Witryna projektu Iperf, http://sourceforge.net/projects/iperf/
8. Witryna dystrybucji Debian, http://www.debian.org
9. Witryna projektu Xen, http://xen.org/
10. Strona projektu Wireshark, http://www.wireshark.org/

# (Nie)docenione rozwiązanie – routing za pomocą protokołu DHCPv6 w środowisku IPv6

Krzysztof Nowicki[1], Mateusz Ożga[1], Tomasz Mrugalski[2]

[1] Katedra Teleinformatyki, Politechnika Gdańska
[2] Internet Systems Consortium

**Streszczenie** W sieciach IPv6 minimalne parametry routingu (adres routera, prefiks sieci oraz metryka trasy) przekazywane są przez mechanizmy protokołu Neighbor Discovery. W pewnych scenariuszach obecnie dostępne mechanizmy w sieciach IPv6 nie są wystarczające do konfiguracji routingu. Grupa robocza Multiple Interfaces (MIF), należąca do IETF, zaproponowała nowy mechanizm dostarczania parametrów routingu [6] za pomocą protokołu DHCPv6. W oparciu o ten dokument powstała pierwsza autorska implementacja tego nowatorskiego rozwiązania. W pracy omówiono możliwości funkcjonalne protokołu DHCPv6, przykłady scenariuszy, w których mechanizmy Neighbor Discovery nie są wystarczające do zapewnienia routingu oraz proponowane w drafcie[7] rozwiązanie. Opisano też autorską implementację mechanizmu proponowanego rozwiązania i wyniki testów.

## 1 Cechy protokołu DHCPv6

Protokół DHCP został stworzony z myślą o szybkiej, sprawnej oraz zautomatyzowanej konfiguracji urządzeń sieciowych, głównie hostów. Dostarczane parametry konfiguracyjne pomagają prawidłowo skonfigurować interfejs sieciowy urządzenia. Oczywiście bez pomocy protokołu DHCP można przeprowadzić konfigurację interfejsu, ale na większą skalę nie jest to praktyczne, zajmuje dużo czasu i jest podatne na błędy.

Protokół DHCP w wersji 6 (DHCPv6) [18] zachowuje się podobnie do swojego poprzednika - protokołu DHCP w wersji 4 (DHCPv4). Podstawowa funkcjonalność, czy też idea, nie zmieniła się. Różnice wynikają z samego środowiska IPv6: stosowana jest inna adresacja oraz istnieją inne mechanizmy rozróżniania sąsiednich węzłów. Typ komunikacji zmienił się na multicast. Z formalnego punktu widzenia DHCPv6 jest protokołem warstwy 3, natomiast DHCPv4 działa na pograniczu warstw 2 i 3 (do wysyłania wielu wiadomości DHCPv4 musi stosować pewne metody obejścia w celu ominięcia protokołu ARP). Wspierany jest też nowy mechanizm nieznany w IPv4 – delegacja prefiksów.

W porównaniu do DHCPv4, w którym jeden komunikat DHCPREQUEST realizował kilka funkcji (żądanie od serwera parametrów adresowych, potwierdzenie ważności przydzielonego adresu sieciowego, przedłużenie dzierżawy danego adresu sieciowego), w nowszej wersji protokołu DHCP wykorzystywane są cztery

komunikaty: REQUEST, CONFIRM oraz RENEW lub REBIND. Istnieje również możliwość uzyskania wyłącznie innych parametrów niż adres lub prefiks, nie przechodząc całego mechanizmu pozyskiwania adresu wraz z parametrami dodatkowymi. Do tego celu został zdefiniowany komunikat INFORMATION-REQUEST, jako odpowiednik wiadomości DHCPINFORM w DHCPv4. Nie ma potrzeby ponownie wyszukiwania serwera DHCP oraz proszenia o wszystkie parametry konfiguracyjne. Do tego wystarcza wymiana tylko dwóch wiadomości pomiędzy stacją kliencką a serwerem. Przykładem takich parametrów są adresy serwerów DNS oraz NTP. Wymienione zostają tylko dwie wiadomości zamiast czterech. Definiowanie swoich opcji jako parametrów dodatkowych też jest możliwe. Wymagane jest zdefiniowanie ich w konfiguracji po stronie serwera oraz klienta.

Nową funkcjonalnością protokołu DHCPv6 jest mechanizm Prefix Delegation. Całość opiera się na przydziale nie tylko pojedynczego adresu, lecz całej puli adresów. Przesyłane prefiksy sieci są wykorzystywane dalej przez urządzenia do konfiguracji pojedynczych hostów. Możliwy jest dalszy podział segmentu na mniejsze podsieci. Mechanizm ten najczęściej jest stosowany pomiędzy routerem PE oraz routerem CE, odpowiednio jako serwer DHCPv6 oraz klient DHCPv6.

W wersji 4 protokołu DHCP możliwe jest dostarczenie hostowi adresu bramy głównej. Na podstawie tej informacji host dodaje do tablicy routingu domyślną trasę, dzięki czemu będzie wiedział gdzie wysyłać pakiet kierowany do danego adresu czy też sieci. Informacja o bramie głównej w protokole DHCPv6 została pominięta. Niemożliwe jest uzyskanie tej informacji poprzez protokół w wersji 6. Wynika to z budowy oraz funkcjonowania środowiska IPv6. Inny zestaw mechanizmów jest odpowiedzialny za konfigurację routingu, w tym przypisanie adresu bramy głównej. Mowa tutaj o protokole Neighbor Discovery i jego mechanizmie wykorzystującym wiadomość Router Advertisement.

Protokół DHCPv6 jest scentralizowany, a zatem jego funkcjonowanie jest zależne od jednego, konkretnego elementu w sieci. Gdy ten element będzie niedostępny z powodu awarii lub w wyniku zamierzonego szkodliwego działania, sieć wspomagana protokołem DHCPv6 będzie z czasem traciła informacje o konfiguracji adresowej hostów. Należy zwrócić uwagę, że identyczny problem występuje w przypadku środowiska IPv4, gdzie awaria scentralizowanego serwera DHCPv4 powoduje podobny brak dostępności usług. Prowadzone są obecnie prace nad rozwiązaniem tego problemu przez definicję mechanizmu failover dla DHCPv6 [1, 2].

## 2 Ograniczenia mechanizmów protokołu Neighbor Discovery

Mechanizmy protokołu Neighbor Discovery (ND) w środowisku IPv6 są wykorzystywane przez każdy węzeł, niezależnie od tego czy jest nim host czy router (wyjątkiem są wiadomości Router Advertisement jednego z mechanizmów protokołu ND, które do niedawna były ignorowane przez routery, ostatnio w pewnych przypadkach są one akceptowane (wymogi WAA-2 i WAA-6, RFC6204 [14]). Ich

zadaniem, między innymi, jest dostarczanie informacji o urządzeniach w najbliższym otoczeniu. Mają na celu określenie, którzy z sąsiadów dostępni są na tym samym łączu (tzw. on-link determination) , wyszukiwanie adresów sieciowych routerów oraz pozyskiwanie od nich dodatkowych parametrów konfiguracyjnych, a także odświeżanie informacji o dostępności ścieżek do danych sąsiednich węzłów.

Do realizacji zadań ND wykorzystywane są wiadomości Neighbor Solicitation i Neighbor Advertisement związane z odkrywaniem sąsiednich węzłów oraz Router Solicitation (RS) i Router Advertisement (RA) odpowiedzialne za wyszukanie routerów i pozyskanie od nich parametrów konfiguracyjnych. Istnieje też mechanizm przekierowania przesyłanych danych - wykorzystywana jest do tego wiadomość Redirect. Polega on na informowaniu hostów o lepszej trasie do punktu docelowego.

Pakiet IPv6 zawierający wiadomość RA jest okresowo rozgłaszany w sieci przez routery. Może on być wysłany jako odpowiedź na wiadomość RS hosta, gdy wiadomość RS będzie prośbą o natychmiastowe przesłanie wiadomości RA, a co za tym idzie prośbą o parametry konfiguracyjne. Wiadomość RA zawiera listę prefiksów do określonych sieci oraz adresy poprzez które sieci te są osiągalne. Każdy host, który otrzymał wiadomość RA, pobiera jej zawartość a następnie uzupełnia swoją tablicę routingu. Tablica ta jest używana podczas podejmowania decyzji kiedy odbiorca pakietu znajduje się bezpośrednio na łączu (on-link), a kiedy jest osiągalny poprzez dany router. Mechanizm wykorzystujący wiadomości RS oraz RA ma na celu pomoc w budowaniu oraz zarządzaniu taką tablicą. Możliwe jest dostarczenie innych parametrów konfiguracyjnych przez wspomniany mechanizm, przykładem jest tutaj konfiguracja adresowa typu stateless. Host nie musi używać serwera DHCPv6 do konfiguracji adresowej interfejsów, może to zrobić autonomicznie, o ile administrator sieci na to zezwolił.

Używając mechanizmów protokołu ND jesteśmy w stanie dostarczyć wymaganą informację o trasach tylko do sąsiednich węzłów lub sieci. Niestety, w wielu przypadkach istniejące mechanizmy ND nie są wystarczające i wymagają odpowiednich rozszerzeń. I tak np. w przypadku, gdy host będzie posiadał więcej tras do danej sieci lub hosta, pojawia się problem wyboru najlepszej trasy, w sensie ścieżki o lepszej metryce. Opracowano rozszerzenia dla mechanizmów ND (dokument RFC 4191, [3]) pozwalające na zdefiniowanie preferencji wyboru trasy lub sprecyzowania niestandardowych tras od routera do hosta.

Jednak dla wielu scenariuszy konfiguracji środowiska IPv6 rozwiązanie problemów routingu za pomocą ND jest nadal bardzo trudne lub niemożliwe:

– Parametrów konfiguracyjnych, związanych zarówno z adresacją jak i routingiem, nie można dostarczać do określonego hosta, a jedynie do wszystkich hostów w sieci – brak jest możliwości zróżnicowania konfiguracji na poszczególne hosty. Wynika to z faktu, że wiadomości RA wysyłane są na adres multicastowy, do wszystkich węzłów. Przykładem może być sieć, w której zwykli użytkownicy komunikują się za pomocą jednego routera, gdy użytkownicy uprzywilejowani wykorzystują inny. Chcąc odseparować konfigurację hostów, nakłonić do korzystania z jednej określonej ścieżki czy wskazać

konkretny router do komunikacji, konfigurację należy przeprowadzić ręcznie na każdym z hostów.
- W środowisku multi-homing, węzeł będący hostem czy też bramą główną, ma do rozwiązania trzy problemy: wybór adresu źródłowego [15], wybór adresu serwera DNS [16] oraz wybór adresu next-hop. O ile dwa pierwsze mogą być rozwiązane przy pomocy protokołu DHCPv6 (prace nad rozwiązaniem wszystkich trzech problemów są prowadzone, całość opisana jest w dokumencie [4]), o tyle trzeci problem nie jest łatwy w rozwiązaniu za pomocą dostępnych mechanizmów protokołu ND. Węzeł będąc w środowisku multi-homing posiada kilka łącz do różnych operatorów. Każdy router w sieci użyje wiadomości RA do dystrybucji domyślniej trasy lub informacji o adresie next-hop, dla hosta lub bramy sieciowej. W tym przypadku, host lub brama sieciowa może wybrać każdą z obowiązujących tras do domyślnego routera. W rezultacie ruch będzie wysłany do niewłaściwego routera i odrzucony przez operatora znajdującego się na użytej trasie. Rozwiązaniem problemu trzeciego mogło by być wprowadzanie możliwości konfiguracji z uwzględnieniem preferencji konfiguracyjnych tras. Dodatkowym atutem jest możliwość zróżnicowanej konfiguracji routingu dla wybranych węzłów.
- Jeśli w danej wydzielonej sieci nie występuje żaden router, hosty nie będą mogły wykorzystać mechanizmu związanego z wiadomościami RS oraz RA. Przypuśćmy, że dwa hosty otrzymują adresy globalne od serwera DHCPv6 za pomocą protokołu DHCPv6 i w sieci nie występuje żaden router. Pomimo, że hosty posiadają przypisany adres o tym samym prefiksie, nie mogą komunikować się ze sobą. Nie posiadają informacji o trasie, poprzez którą mogą być osiągalne. Sytuacja taka może występować np. w sieciach separowanych, w których uplink dostępny jest tylko okresowo (np. cykliczny odczyt sensorów raz na kilka godzin).
- W statycznych sieciach, takich jak sieci korporacyjne lub nawet domowa sieć z jednym routerem, możliwe jest niekorzystanie z mechanizmu operującego wiadomościami RS oraz RA i dostarczenie parametrów konfiguracyjnych używając wyłącznie protokołu DHCPv6. Konfiguracja ta rozwiązuje problem fałszywych wiadomości RA, które mogą być z łatwością wysyłane do sieci przez napastnika w celu manipulacji konfiguracją węzłów. Jeśli istnieje związek zabezpieczeń (ang. Security Association), czyli sposób korzystania z usług ochrony podczas komunikacji pomiędzy klientem a serwerem DHCPv6, może być opłacalne wykorzystanie protokołu DHCPv6 oraz wyłączenie wsparcia mechanizmu opartego o wiadomości RS i RA.
- Użycie mechanizmu RA jest w kłopotliwe w sieciach, które nie oferują rozgłaszania pakietów. Ta grupa technologii dostępowych określana jest jako Non-Broadcast Multiple Access (NBMA). Przykładem takiej technologii mogą być sieci komórkowe UMTS, gdzie dla każdego terminala zestawiane są oddzielne tunele.

W mechanizmie wykorzystującym wiadomości RS oraz RA nie istnieje ściśle zdefiniowany scentralizowany system zarządzania prefiksami oraz opcjami związanymi z routingiem. Rolę takiego systemu może zapewne pełnić DHCPv6. Wpro-

wadzanie opcji routingu do protokołu DHCPv6 może rozwiązać nie tylko powyższe problemy, ale także wiele innych. Funkcjonalność ta daje nam o wiele wygodniejsze konfigurowanie węzłów oraz możliwość dostarczenia wszystkich niezbędnych parametrów konfiguracyjnych, jak dotąd niedostępnych w protokole DHCP wersji 6.

## 3 Propozycja konfiguracji routingu z użyciem protokołu DHCPv6

Mechanizm przenoszenia informacji o routingu za pomocą protokołu DHCPv6 powstał w ramach stowarzyszenia IETF. Grupa robocza o nazwie MIF (Multiple Interfaces), działająca w obszarze internetowym stowarzyszenia, zaproponowała dokument w postaci draftu (ang. Internet-Draft, [5]). Draft draft-ietf-mif-dhcpv6-route-option-05 [7] opisuje dokładnie zachowanie mechanizmu, definiuje dostarczane parametry oraz ograniczenia proponowanego rozwiązania. Współautorem dokumentu jest Tomasz Mrugalski, jeden z autorów niniejszego artykułu. Brał on udział w pracach MIF, która rozwija i ulepsza proponowane rozwiązanie.

Gotowość do przyjęcia parametrów routingu przez klienta DHCPv6 jest zgłaszana za pomocą opcji Option Request Option (ORO) w wiadomościach DHCPv6. ORO jest używane do określenia listy parametrów, które będą transmitowane pomiędzy klientem a serwerem. Klient wykorzystując ORO ma możliwość żądania konkretnych parametrów od serwera DHCPv6. Może on dołączyć opcję ORO, wraz z listą interesujących go parametrów, w zapytaniach do serwera używając wiadomości SOLICIT, REQUEST, RENEW, REBIND oraz INFORMATION-REQUEST. Serwer powinien dostarczyć żądane parametry, dla każdej pozycji na liście, o które proszony jest przez klienta. Tylko komunikaty ADVERTISE oraz REPLY, wysyłane przez serwer DHCPv6, będą zawierały opcje żądanie przez klienta.

Aby skonfigurować trasę do danej sieci należy posiadać informacje o adresie next-hop, prefiksie sieci oraz czasie obowiązywania trasy (lifetime). Informacje te przekazywane są w opcji Next Hop Option oraz podopcji Route Prefix Option:

- Next Hop Option jako opcja NEXT_HOP - definiuje adres next-hop, zazwyczaj odpowiadający specyficznemu adresowi next-hop routera. Dla każdego adresu next-hop, może być zero, jeden lub więcej prefiksów osiągalnych poprzez ten adres. Każdy wpis o routingu zawiera adres next-hop, prefiks docelowy (docelowa podsieć) oraz wartość preferencji dla trasy. Elementy te dostarczane są w opcji NEXT_HOP za pomocą podopcji RT_PREFIX.
- Route Prefix Option jako podopcja RT_PREFIX - jest używana do dostarczania informacji o pojedynczym prefiksie, który reprezentuje sieć docelową. Zdefiniowana jako pod-opcja NEXT_HOP. Może być wysłana bezpośrednio w wiadomości by wskazać, że dana trasa jest bezpośrednio na łączu.

Ważnym parametrem przypisanym do podopcji RT_PREFIX jest parametr Lifetime. Oznacza on czas, przez jaki dany wpis o routingu obowiązuje. Jeśli serwer wyśle parametr z lifetime równym 0, stacja kliencka powinna dany wpis

usunąć ze swojej tablicy routingu. To samo powinna zrobić w przypadku gdy lifetime osiągnie wartość 0. Wartość lifetime może przybrać wartość nieskończoności, wtedy wpis o danej trasie będzie obowiązywał dopóki serwer nie ustawi wartości na 0 lub wartość ta przybierze liczbę skończoną. Wartość preferencji dla trasy jest zapisana na dwóch bitach, więc definiują ją cztery wartości:

```
01 - wysoki priorytet
00 - normalny priorytet (wartość domyślna)
11 - niski priorytet
10 - zarezerwowane
```

Wartości te zostały dobrane w taki sposób, aby zachować kompatybilność z RFC 4191.

Za pomocą przywołanych parametrów routingu możliwe jest skonfigurowanie tras:

- specyficznej – dana sieć jest osiągalna poprzez dany router,
- default (domyślnej) – domyślna trasa jest dostępna poprzez dany router,
- on-link – dana sieć jest bezpośrednio na łączu on-link.

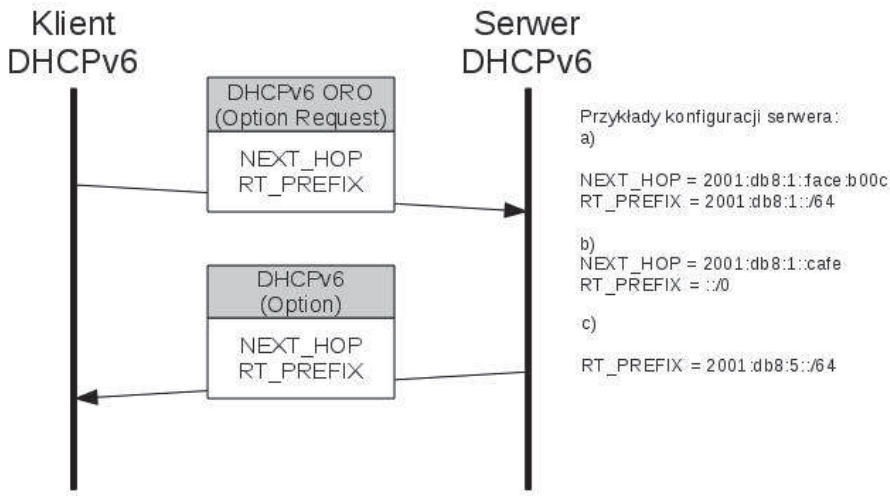
Proponowany mechanizm dostarczania opcji routingu za pomocą protokółu DHCPv6 sprowadza się do wymiany dwóch wiadomości między klientem a serwerem. Klient zainteresowany parametrami dotyczącymi routingu powinien wysłać prośbę do serwera zawierającą ustawione opcje NEXT_HOP i RT_PREFIX. Zadaniem serwera jest dostarczenie parametrów w zależności od jego konfiguracji. Poniżej znajdują się przykłady pozyskiwania tras poprzez proponowany mechanizm.

Na rysunku 1 przedstawiono przykład komunikacji między klientem a serwerem DHCPv6, za pomocą której odbywa się pozyskiwanie parametrów routingu. W zależności od konfiguracji serwera, która jest przedstawiona w punktach a), b) i c) na rysunku, dostarczane są odpowiednie trasy. Przy pomocy pierwszej konfiguracji (punkt a)) serwera DHCPv6, dostarczana jest informacja o trasie specyficznej. Klient wysyła wiadomość zawierającą ORO z żądaniem o parametry routingu. Serwer w odpowiedzi na żądanie, w zależności od konfiguracji, dostarcza parametry do klienta. W tym przypadku będą to parametry odnośnie trasy specyficznej. Opcja NEXT_HOP zawiera adres danego routera, a pod-opcja RT_PREFIX sieć, która jest osiągalna przez dany router. Parametr lifetime jest przekazywany w podopcji RT_PREFIX. Po udanej wymianie wiadomości, stacja kliencka powinna posiadać wpis o trasie:

```
sieć 2001:db8:1::/64 osiągalna poprzez adres 2001:db8:1::face:b00c.
```

Za pomocą konfiguracji drugiej (punkt b)) możliwe jest pozyskanie informacji o trasie domyślnej. Stacja kliencka wysyła prośbę o parametry routingu w opcji ORO. Serwer, w tym przypadku, odpowiada przesyłając parametry dla ścieżki domyślnej, umieszczając pod-opcję RT_PREFIX z prefiksem ::/0. W wyniku wymiany wiadomości o takich parametrach, stacja kliencka będzie posiadać wpis o trasie domyślnej:

```
default via 2001:db8:1::cafe.
```



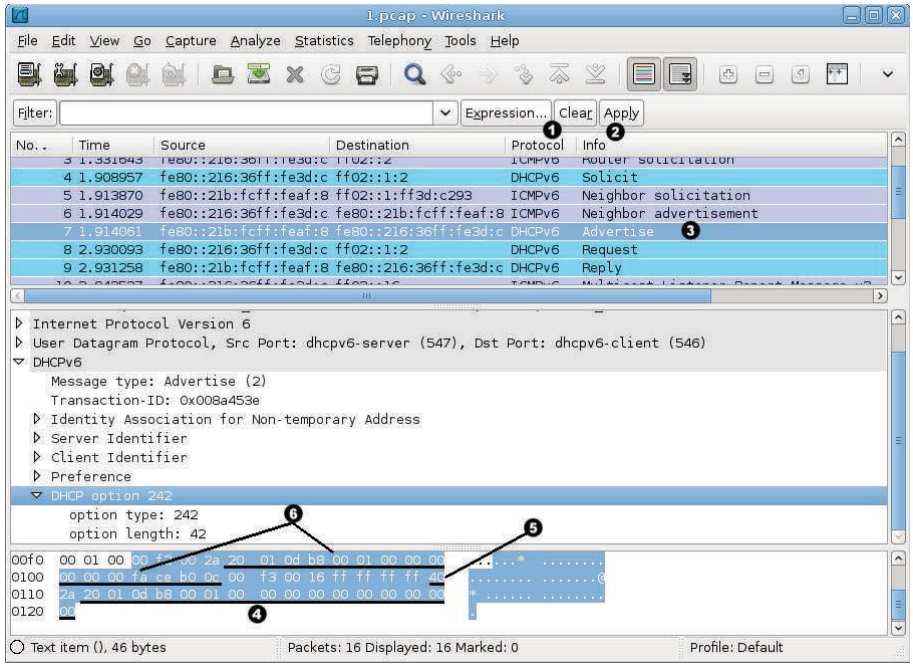

**Rysunek 1.** Przykład komunikacji między klientem a serwerem DHCPv6

Konfiguracja trzecia (punkt c)) serwera DHCPv6 definiuje trasę on-link. Tak jak w poprzednich przykładach, stacja kliencka prosi serwer DHCPv6 o parametry routingu. Serwer posiadając tylko wpis o trasie on-link, odpowiada komunikatem z żądanymi opcjami przez klienta. W tym przykładzie konfigurowana jest trasa on-link, więc pod-opcja RT_PREFIX może być zawarta bezpośrednio w wiadomości, bez opakowania jej w opcję NEXT_HOP. Stacja kliencka po otrzymaniu odpowiednich parametrów będzie mogła dodać wpis o trasie on-link.

Na rysunku 2 znajduje się zrzut ekranu aplikacji Wireshark (sniffer sieciowy), która nasłuchiwała komunikację między klientem a serwerem DHCPv6. W oknie zebranych pakietów prezentowane są wiadomości uczestniczące w komunikacji. Sekcja Protocol (1) pokazuje typ protokołu, a sekcja Info (2) mówi o typie wiadomości. Należy zwrócić uwagę na pakiety z wpisem DHCPv6 w sekcji Protocol. Wymiana pakietów o takich wiadomościach definiuje mechanizm konfiguracji adresowej klienta.

Zaznaczony pakiet (3), o typie wiadomości ADVERTISE, zawiera żądane przez klienta DHCPv6 parametry routingu. Drugie okno aplikacji zawiera strukturę pakietu, a w trzecim oknie widać konkretne wartości związane z opcjami routingu. W tym przypadku przesyłane są informacje o trasie specyficznej, wartości te są zgodne z przykładem konfiguracji a) rysunku nr 1, sieć 2001:db8:1::/64 jest osiągalna poprzez adres 2001:db8:1::face:b00c. Na rysunku wyszczególniony jest adres sieci (4) wraz z wielkością prefiksu (5) oraz adres routera (6). Wartości te prezentowane są w systemie szesnastkowym.

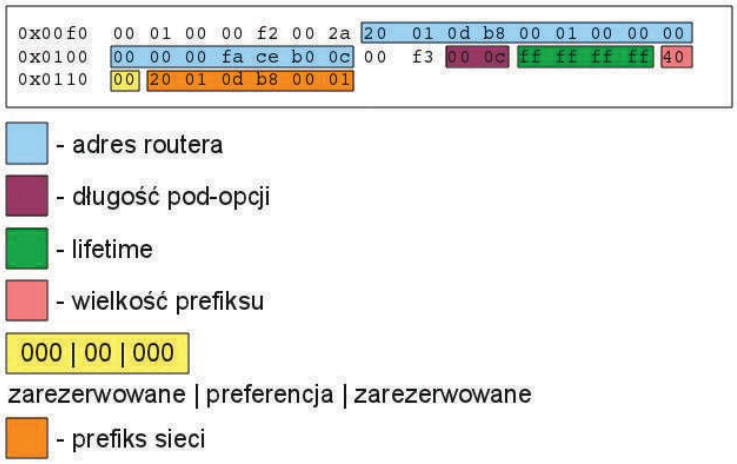
Przytoczone przykłady konfiguracji serwera obrazują sposób konfiguracji pojedynczych przypadków czy też tras. Serwer może dostarczać wszelkie możliwe parametry, nie sprowadza się to do przesyłania tylko jednego parametru. Jeśli




**Rysunek 2.** Aplikacja Wireshark – parametry routingu w komunikacji między klientem a serwerem DHCPv6

serwer będzie posiadał wiele wpisów o trasach specyficznych lub on-link, oraz będzie tak skonfigurowany by dostarczyć te parametry, na prośbę klienta w odpowiedzi prześle parametry wykorzystując opisywany mechanizm. Możliwe jest rozróżnienie ze względu na klienta, przekazanie innych parametrów do każdego hosta oraz rozróżnienie ze względu na interfejs klienta, różne parametry routingu na różnych interfejsach sieciowych.

Zgodnie z najnowszą wersją draftu [7], opcja w wiadomości DHCPv6 prawdopodobnie będzie posiadała postać jak na rysunku 3. W porównaniu z wersją 03 draftu zmiany dotyczą tylko formatu opcji, dokładnie występujących w nich parametrów. Cała idea dostarczania parametrów routingu przez protokół DHCPv6 pozostaje bez zmian. Na rysunku 3 wyszczególnione są odpowiednie parametry routingu dostarczane przez mechanizm. Tak, zgodnie z wersją 05 draftu, mógłby wyglądać format wiadomości dla przypadku a) z rysunku 1. Należy zwrócić uwagę na długość podopcji (kolor fioletowy), która jest zmienna i zależna od prefiksu sieci (kolor pomarańczowy). Zyskujemy tutaj na długości całego pakietu. Następną zmianą jest dwubitowe pole preferencji (kolor żółty), które definiuje preferencję dla trasy. W przypadku lifetime, wartość ta jest ustawiona na wartość nieskończoności (0xffffffff).




**Rysunek 3.** Format wiadomości zgodnie z draftem w wersji 05

## 4 Implementacja i testy

Zgodnie z zaleceniami dokumentu draft w wersji 03 [6], opisującego mechanizm dostarczania parametrów routingu, została stworzona implementacja jako rozszerzenie dla aplikacji Dibbler [8]. Dibbler jest to implementacja klienta, serwera oraz przekaźnika DHCPv6. Jest ona rozpowszechniana na zasadach licencji GNU GPL, wiec jej użytkowanie jest nieograniczone, również w zastosowaniach komercyjnych. Dibbler działa na różnych platformach, do dyspozycji są systemy z rodziny Windows, GNU/Linux, Mac OS X, a także kilka członków rodziny BSD. Obsługuje wiele architektur, m.in. i386. x64, Sparc, PowerPc, MIPS oraz ARMv5. Wspiera on zarówno konfigurację stateful jak i stateless stacji klienckich. Protokół DHCPv6 definiuje ponad 78 opcji dla konfiguracji stacji klienckich, większość z nich jest obsługiwana przez Dibblera. Mechanizm dostarczania opcji routingu został dodany do wersji Dibblera o numerze 0.8.1 (RC1) i jest dostępny we wszystkich późniejszych wydaniach.

Implementacja mechanizmu dostarczania parametrów routingu została napisana w języku C++ jako opcja dla DHCPv6. Konfiguracja parametrów routingu odbywa się za pomocą pliku konfiguracyjnego serwera DHCPv6. Również w tym pliku znajduje się konfiguracja całego serwera, sposób funkcjonowania, konfiguracja adresowa hostów czy rozróżnienie konfiguracji na interfejsy sieciowe. Serwer posiadając odpowiednie wpisy na temat tras będzie mógł odpowiadać na zapytania o parametry routingu. Listing 1 zawiera fragment pliku konfiguracyjnego serwera, w którym znajdują się wpisy na temat routingu dla wszystkich konfiguracji tras z rysunku nr 1. W liniach 15-17 znajduje się definicja trasy domyślnej. Sekcja pomiędzy liniami 19 a 23 zawiera informację o trasie specyficznej z lifetime równym nieskończoność. Linia 25 pokazuje przykład zdefiniowania trasy on-link z lifetime wynoszącym 3600.

```
iface "eth0" {

 // also ranges can be defines, instead of exact values
 t1 1800-2000
 t2 2700-3000
 prefered-lifetime 3600
 valid-lifetime 7200

 # assign addresses from this pool
 class {
   pool 2001:db8:1::/64
 }

 # a single next-hop without any routes defined (i.e. default router)
 next-hop 2001:db8:1::cafe {
   route ::/0
 }

 # router with a single route with infinite lifetime
 next-hop 2001:db8:1::face:b00c {
   route 2001:db8:1::/64
 }

 # prefixes available on link directly, not via router
 route 2001:db8:5::/64 lifetime 3600
 }
```

Listing 1: fragment pliku konfiguracyjnego serwera DHCPv6 zawierający parametry routingu dla różnych tras.

Po stronie klienta istnieje tylko jeden wpis w pliku konfiguracyjnym aplikacji. Mówi on o tym czy klient ma żądać parametrów dotyczących routingu czy nie. Wpis taki może pojawić się w każdej sekcji interfejsu sieciowego w pliku konfiguracyjnym. Listing 2 przedstawia przykład pliku konfiguracyjnego klienta DHCPv6. Linia 6 w tym przypadku informuje by aplikacja klienta DHCPv6 żądała parametrów routingu od serwera DHCPv6.

```
1 script "/etc/dibbler/client-notify.sh"

3 iface "eth0" {
   ia
5
   routing 1
7 }
```

Listing 2: przykład pliku konfiguracyjnego klienta DHCPv6 z ustawioną opcją żądania o parametry routingu.

Po otrzymaniu parametrów przez klienta, do ustawienia tras w systemie, użyto skryptu konfiguracyjnego. Na listingu 3 znajduje się fragment skryptu powłoki, który w zależności od występowania parametrów, dokonuje odpowiednich wpisów w tablicy routingu. W tym przypadku jest to ustawienie trasy domyślnej. Opcje routingu są przekazywane do skryptu jako zmienne środowiskowe. Cały proces pozyskiwania parametrów oraz tworzenia wpisów o routingu może być raportowany do pliku z logami.

```
1 ...
  if [ "$OPTION_NEXT_HOP" != "" ]; then
3    ip -6 route del default > /dev/null 2>&1
     ip -6 route add default via ${OPTION_NEXT_HOP} dev $IFACE
5 ...
```

Listing 3: fragment skryptu wspomagającego ustawienie routingu po stronie klienta.

Testy mechanizmu były przeprowadzone w systemach opartych o dystrybucję Debian (GNU/Linux) z wersją jądra 2.6.3x. Środowisko testowe zawierało dwa kluczowe elementy, hosta z uruchomianą aplikacją Dibbler w roli klienta DHCPv6 oraz hosta w roli serwera DHCPv6, na którym również działał Dibbler. W środowisku znajdował się jeszcze trzeci element. Był nim router, który zapewniał dostęp do sieci Internet wykorzystując do tego protokołu IPv6. Przyjęto, że wszystkie te elementy są połączone do tej samej sieci.

W konfiguracji klienta DHCPv6 znajdował się wpis mówiący o pozyskiwaniu adresu sieciowego od serwera DHCPv6. Dodatkowym wpisem była prośba o parametry routingu od serwera DHCPv6. Serwer DHCPv6 w swojej konfiguracji posiadał informację o puli adresowej, z której miał przydzielać adresy dla klientów oraz adres serwera DNS (został użyty publiczny serwer DNS, by nie wprowadzać dodatkowych elementów do środowiska testowego). W konfiguracji zostały też uwzględnione czasy dzierżawy dla adresów. W kolejnych liniach pliku konfiguracyjnego serwera DHCPv6 znajdowały się wpisy o opcjach routingu. Ponieważ testowane było przekazywanie różnych parametrów routingu, pod względem ilości opcji oraz typu trasy, za każdym razem wpisy te były inne. Całe środowisko, a przede wszystkim router, nie korzystało z mechanizmów opartych o wiadomości RS i RA. Podczas przeprowadzania testów na aplikacjach Dibbler włączony został maksymalny poziom logowania.

Testy polegały na sprawdzaniu czy host klienta DHCPv6 otrzymuje parametry routingu zgodne z ustawionymi po stronie serwera DHCPv6 oraz czy tworzy wpisy w tablicach routingu. Zaraz po konfiguracji adresowej interfejsu sieciowego klienta DHCPv6 powinna zacząć się konfiguracja routingu na hoście klienta. By potwierdzić działanie mechanizmu, oprócz przeglądania tablicy routingu hosta, użyto systemowego polecenia ping. Zapytania przy pomocy polecenia ping były kierowane do adresu bramy głównej, adresu hosta w innej sieci oraz adresu publicznego w sieci Internet. Jeśli klient DHCPv6 posiadał wpisy na temat tras do danych adresów, gdzie wpisy były zależne od konfiguracji serwera DHCPv6, to otrzymywał odpowiedzi na zapytania polecenia ping.

Zostało przetestowane pozyskiwanie informacji o routingu o trasie domyślnej, on-link oraz specyficznej. Nie ograniczano się tu do przekazywania tylko jednej trasy, testowane były różne możliwe warianty pod względem liczby oraz typu tras. Na potrzebę testów dla trasy specyficznej do środowiska został dodany router oraz kolejny host znajdujący się w innej sieci (konfigurację adresową tego hosta wykonano ręcznie).

W początkowych fazach testowania implementacji wykryto kilka drobnych błędów. Adresy hostów oraz prefiksy sieci były nieprawidłowo przetwarzane

przez serwer DHCPv6, a co za tym idzie po wysłaniu do klienta DHCPv6 pojawiał się problem z odczytaniem. Problem ten został poprawiony.

Mechanizm przesyłania parametrów routingu za pomocą DHCPv6 (na podstawie draftu wersji 03 [6]) jest aktualnie zaimplementowany również w aplikacji ISC DHCP [9]. Tak samo jak Dibbler, jest ona darmową otwarto-źródłową (udostępnianą na zasadach licencji BSD) implementacją klienta, serwera oraz przekaźnika DHCP. Działa w środowisku IPv4 oraz IPv6 i jest używana na szeroką skalę. Zazwyczaj jest dostarczana jako standardowy klient DHCP w systemach uniksopodobnych [10].

Testy implementacji zostały również przeprowadzone pomiędzy aplikacją Dibbler oraz aplikacją ISC DHCP. Testowano różne konfiguracje, zmieniały się zarówno dostarczane parametry jak i role aplikacji (klient-serwer, serwer-klient). Wyniki testów były pozytywne, obie implementacje przekazywały parametry routingu między sobą zgodnie z założeniami mechanizmu.

W wersji 0.8.3 Dibblera dostępne są skrypty konfigurujące routing na systemach OpenBSD. Możliwe jest jego wykorzystanie na innych systemach z rodziny BSD, choć może to wymagać kosmetycznych zmian w skrypcie powłoki.

## 5 Proces standaryzacji

Dokumenty Internet-Drafts [5] są serią dokumentów roboczych publikowanych przez IETF, które w domyśle dążą i są kreowane do uzyskania statusu standardu RFC. Dokument Internet-Draft jest ważny przez 6 miesięcy od wydania, chyba że zostanie zastąpiony nowszą wersją tej samej specyfikacji. Czas wygaśnięcia zmieni się wraz z wydaniem nowszej wersji dokumentu, od nowa będzie liczone 6 miesięcy. Początkowo drafty tworzone są przez osoby indywidualne (tzw. individual submission). W przypadku zainteresowania społeczności są one akceptowane przez jedną z grup roboczych (tzw. adoption) i od tego czasu określane są jako dokumenty całej grupy roboczej. W pewnym stopniu zaawansowania ogłaszane jest ostateczne wezwanie do komentarzy (tzw. Work Group Last Call lub WGLC). Po jego pomyślnym zakończeniu draft przekazywany jest do grupy Internet Engineering Steering Group (IESG). Dostępna jest również inna ścieżka publikacyjna, kiedy to z ważnych względów dokument pomija fazę grupowa i trafia od razu do IESG . Grupa ta jest częścią IETF i zajmuje się finalnym technicznym przeglądem internetowych standardów oraz podejmuje decyzje w sprawie akceptacji dokumentów by stały się normami. Przegląd grupy IESG następuje wtedy, gdy zgłosi się prośbę by dany dokument opublikować jako dokument RFC.

Trzeba mieć na uwadze, że dokumentu Internet-Draft nie można traktować jako publikowanej specyfikacji. Nie posiada on formalnego statusu i może ulec zmianie lub wykluczeniu w dowolnym momencie. Dlatego nie powinien być podawany lub cytowany w żadnym oficjalnym dokumencie w sposób inny niż prace w toku (ang. work-in-progress).

Opis proponowanego mechanizmu dostarczania opcji routingu znajduje się w drafcie draft-ietf-mif-dhcpv6-route-option-05 [7]. Dokument ten w przeciągu

1,5 roku ostatecznie osiągnął numer 05. Odnosząc się do innych draftów, nasze rozwiązanie zdobyło duże zainteresowanie i przeszło znaczącą część ścieżki standaryzacyjnej: Uzyskało akceptację grupy roboczej, po czym zostało ogłoszone WGLC. Zazwyczaj drafty prezentujące nową ideę kończą się na numerze draftu 01 lub 02 i dalsze prace nad nimi zostają zaniechane. Istotnym wyznacznikiem popularności prowadzonych prac jest ilość innych draftów odwołujących się do proponowanego rozwiązania [17].

## 6 Konsekwencje wdrożenia mechanizmu dostarczania opcji routingu za pomocą protokołu DHCPv6

Mechanizm dostarczania opcji routingu za pomocą protokołu DHCPv6 rozwiązuje problemy nie tylko specyficznych konfiguracji sieci, ale może być użyty w mniej złożonych przypadkach. Przytoczone przykłady w rozdziale 2 można spotkać na co dzień, zarówno w małych jak i dużych środowiskach sieciowych. Mechanizm ten może idealnie zaspokajać potrzeby tych środowisk. Dzięki niemu niezbędna konfiguracja interfejsów sieciowych do komunikacji między sieciowej staje się łatwiejsza i wygodniejsza.

Po wdrożeniu proponowanego rozwiązania z punktu widzenia użytkownika nic się nie zmienia, bo nadal korzysta on z protokołu DHCP, choćby w celu pozyskania adresów DNS czy nazwy domeny, których dostarczenie za pomocą mechanizmu operującego wiadomościami RS oraz RA staje się problematyczne. Natomiast z punktu widzenia operatora, zmiana jest widoczna. Korzysta on wyłącznie z jednego protokołu, więc zaoszczędza na problemach, które zawsze mogą się pojawić, czy to jest awaria czy błąd w konfiguracji. Operator posiada scentralizowany system nadzorujący prefiksami oraz opcjami routingu, gdzie przy protokole ND mamy tylko konfigurację stateless. Może dokonać konfiguracji tylko określonych stacji klienckich, co daje mu możliwość na wydzielenie konkretnych hostów pod względem wyboru trasy.

Należy pamiętać, że proponowany mechanizm nie działa jako dynamiczny protokół routingu. Nie jest on w stanie sam z siebie wykryć zmian w routingu. Nie zaleca się wykorzystywania go w środowisku, gdzie funkcjonuje dynamiczny routing. Mogą pojawić się wtedy konflikty dotyczące routingu. Możliwością użycia go w takim środowisku jest znalezienie metody na aktualizację zmian w routigu dla serwera DHCP.

Do wad proponowanego rozwiązania dostarczania opcji routingu za pomocą protokołu DHCPv6 należy zaliczyć zwiększoną złożoność sieci. Wykorzystywanie równocześnie dwóch metod zapewniających parametry konfiguracyjne hostom, mechanizm protokołu ND do konfiguracji routingu oraz protokół DHCPv6 do konfiguracji adresowej, zwiększa złożoność sieci. Należy też zwrócić uwagę na fakt, że wprowadzenie dodatkowej metody konfiguracji routingu, zwiększa nakłady konieczne na testowanie produktu, który je wspiera. Oprócz przetestowania obu rozwiązań konieczna jest też walidacja ich wzajemnych interakcji.

## 7 Podsumowanie

Pod koniec marca 2012 roku na 83 spotkaniu IETF w Paryżu [11], miała miejsce kolejna dyskusja na temat proponowanego rozwiązania [12]. Dyskusja sprowadzała się do argumentów dla protokołu RA przeciwko DHCPv6, przytaczając realne przypadki użycia. Jedne z głównych zarzutów przeciwko rozwiązania problemu przy pomocy protokołu DHCPv6 była konfiguracja trasy domyślnej (default route). Uznano, że wycofanie się z możliwości konfiguracji trasy domyślnej byłoby dobrym kompromisem pomiędzy RA a DHCPv6. Niestety takie zmiany spowodowałyby rozwiązanie niemal bezużyteczne w środowisku gdzie dane urządzenie posiada wiele interfejsów sieciowych. Uczestnicy spotkania postulowali, aby do istniejącego problemu routingu w sieciach IPv6 zastosować "jakieś" zupełnie inne podejście, nie oparte o protokół DHCPv6. Być może będzie to nowy protokół dla konfiguracji routerów.

Decyzje w IETF podejmowane są jednak w myśl zasady consensusu, więc nie jest praktykowane głosowanie większościowe. Ze względu na globalny charakter prac IETF nie jest możliwa fizyczna obecność wszystkich członków grup roboczych na jednym spotkaniu, dlatego też wszystkie decyzje muszą być potwierdzone na liście dyskusyjnej.

Grupa standaryzacyjna IETF zwraca dużą uwagę na konsekwencje wprowadzania nowych standardów - muszą one być nie tylko dobre ale zgodne, kompatybilne z istniejącymi rozwiązaniami. W procesie standaryzacji stara się nie wprowadzać zbyt wielu rozwiązań, często kosztem jakości. Nasze rozwiązanie zostało docenione przez projektantów sieci, ale to nie wystarczyło do wejścia na szybką drogę uzyskania statusu dokumentu RFC.

Chociaż wydawało się, że prace nad proponowanym standardem ustaną i nie będą kontynuowane w najbliższym czasie, to jednak duże zainteresowanie rozwiązaniem m.in. przez firmy Telecom Italia, NTT NT Lab z Japonii, organizację standaryzacyjną Broadband Forum, a także przychylność szefa obszaru Internet (tzw. Area Director) spowodowały, że prace są nadal kontynuowane.

Jest także szansa na publikację tego rozwiązania w nieco inny sposób. Dokument RFC2026 [13] mówi o możliwości publikacji draftu o statusie Experimental lub Informational, który z różnych względów nie ma szansy zostać opublikowany tradycyjną ścieżką. Jeśli według stanowiska IESG dany dokument draft proponuje coś faktycznie sprzecznego z istniejącymi pracami, standardami, nadal może uzyskać wspomniane statusy i tym samym nadal może być rozwijany (sekcja 4.2.3, RFC2026 [13]). W takim przypadku IESG może wstawić odpowiednie zastrzeżenia do tekstu dokumentu lub bezpośrednio po sekcji "Status of this Memo", nie może jednak odmówić jego publikacji.

Routing w środowisku IPv6 wymaga dalszych badań. Na dzień dzisiejszy nie istnieje żadna alternatywa lub inne rozwiązania na przytoczone problemy routingu w środowisku IPv6. Spośród dostępnych na rynku implementacji DHCPv6 tylko dwie [8, 10] mogą poradzić sobie z problemem, bowiem wspierają one opcję dostarczania parametrów routingu przy pomocy protokołu DHCPv6.

## Literatura


1. Mrugalski T., Kinnear K.: *DHCPv6 Failover Requirements.* draft-ietf-dhc-dhcpv6-failover-requirements-02 (work-in-progress), IETF, wrzesień 2012.
2. Mrugalski T., Kinnear K.: *DHCPv6 Failover Design.* draft-mrugalski-dhc-dhcpv6-failover-design-02 (work-in-progress), IETF, październik 2012.
3. Draves R., Thaler D.: *Default Router Preferences and More-Specific Routes.* RFC 4191, IETF, listopad 2005.
4. Troan O., Miles D., Matsushima S., Okimoto T., Wing D.: *IPv6 Multihoming without Network Address Translation.* draft-ietf-v6ops-ipv6-multihoming-without-ipv6nat-04 (work-in-progress), IETF,21 luty 2012.
5. Internet-Drafts (I-Ds) (`http://www.ietf.org/id-info/`).
6. Dec W., Mrugalski T., Sun T., Sarikaya B.: *DHCPv6 Route Options.* draft-ietf-mif-dhcpv6-route-option-03 (work-in-progress), IETF, 10 września 2011.
7. Dec W., Mrugalski T., Sun T., Sarikaya B., Matsumoto A.: *DHCPv6 Route Options.* draft-ietf-mif-dhcpv6-route-option-05 (work-in-progress), IETF, 24 sierpień 2012.
8. Dibbler – a portable DHCPv6, strona domowa projektu, `http://klub.com.pl/dhcpv6/`
9. Routing configuration over DHCPv6, publikacja na stronie firmy ISC, (`http://www.isc.org/community/blog/201111/routing-configuration-over-dhcpv6`).
10. ISC DHCP, strona projektu, (`http://www.isc.org/software/dhcp`).
11. IETF 83 - Paris, France (`http://www.ietf.org/meeting/83/index.html`).
12. Prezentacja ze spotkania grupy MIF, IETF83, Paryż, draft-ietf-mif-dhcpv6-route-option-04 (Chairs&Woj/Tomek) (`http://www.ietf.org/proceedings/83/minutes/minutes-83-mif`)
13. Bradner S.: *The Internet Standards Process – Revision 3.* RFC2026. Październik 1996.
14. Singh H., Beebee W., Donley C., Stark B., Troan O.: *Basic Requirements for IPv6 Customer Edge Routers.* RFC6204. Kwiecień 2011.
15. Matsumoto A., Fujisaki T., Chown T.: *Distributing Address Selection Policy using DHCPv6.* draft-ietf-6man-addr-select-opt-08 (work-in-progress), IETF, 16 Styczeń 2013.
16. Savolainen T., Kato J., Lemon T.: *Improved Recursive DNS Server Selection for Multi-Interfaced Nodes.* RFC6731. Grudzień 2012.
17. Draft/RFC Dependency Tool, zależności w innych dokumentach, (`http://www.fenron.net/~fenner/ietf/deps/index.cgi?dep=draft-ietf-mif-dhcpv6-route-option`).
18. Nowicki K., Mrugalski T., Ożga M.: *Wspomaganie routingu przez protokół DHCPv6*, Przegląd Telekomunikacyjny, Wiadomości Telekomunikacyjne, 2012, R. 85, pp. 1362-1371.

# Zasoby i eksperymenty w sieci PL-LABv6

Józef Woźniak[1], Krzysztof Nowicki[1], Tomasz Mrugalski[2], Wojciech Gumiński[1], Henryk Gut[3], Jordi Mongay Batalla[3], Waldemar Latoszek[3], Artur Binczewski[4], Bartosz Gajda[4], Wiktor Procyk[4], Piotr Skałecki[4], Krzysztof Chudzik[5], Jan Kwiatkowski[5], Kamil Nowak[5]

[1] Katedra Teleinformatyki, Politechnika Gdańska
[2] Internet Systems Consortium
[3] Instytut Łączności – Państwowy Instytut Badawczy
[4] Poznańskie Centrum Superkomputerowo-Sieciowe
[5] Politechnika Wrocławska

**Streszczenie** W artykule przedstawiona została sieć PL-LABv6 wraz z zasobami sprzętowymi oraz usługami sieciowymi i aplikacjami. Celem artykułu jest zaznajomienie czytelnika ze specyfikacją sieci widzianą z perspektywy użytkownika-badacza, który nie będąc uczestnikiem projektu IIP, chciałby wykorzystać sieć PL-LABv6 do przeprowadzania własnych badań i eksperymentów.

## 1 Wprowadzenie

Jedną z głównych idei przy tworzeniu sieci eksperymentalnej PL-LAB IPv6 (PL-LABv6) było udostępnienie środowiska testowego na potrzeby badań i eksperymentów prowadzonych głównie w ramach projektu IIP [1], ale również poza nim. Warto podkreślić, iż to zamierzenie - udostępnienie ogólnokrajowej sieci eksperymentalnej IPv6 na potrzeby badań naukowych w Polsce - jest ważną i przełomową koncepcją, która w łatwy i szybki sposób przybliża możliwości dalszego rozwoju oraz testowania wdrożeń protokołu IPv6 w Polsce. Spełnienie tego postulatu wymaga oczywiście zachowania odpowiednich rygorów czasowych, zasobowych oraz organizacyjnych, tak aby nie zakłócało to prac przeprowadzanych w ramach projektu. Oznacza to, ze część zasobów sprzętowych, połączeń sieciowych oraz zainstalowanych usług i aplikacji będzie mogła być udostępniana na potrzeby realizacji badań na zewnątrz projektu.

## 2 Zasoby oraz usługi sieciowe w laboratoriach

Ogólnopolska sieć eksperymentalna PL-LABv6 powstała w oparciu o sprzęt komputerowy (serwery), sieciowy (routery, przełączniki, punkty dostępowe sieci bezprzewodowej i inne) oraz połączenia sieciowe (realizowane lokalnie w ramach laboratoriów w poszczególnych miastach oraz ogólnopolskie – w oparciu o infrastrukturę sieci PIONIER). Łączność pomiędzy laboratoriami poprzez sieć PIONIER realizowana jest na poziomie sieci Ethernet w oparciu o etykiety MPLS

i usługę VPLS (ang. Virtual Private LAN Service), która jest konfigurowana i zarządzana w ramach sieci PL-LAB, utworzonej na potrzeby całego projektu IIP [1]. W niniejszym artykule przedstawiono część zasobów i funkcjonalności wykorzystywanych na potrzeby protokołu IPv6 i dedykowanej do tego celu sieci PL-LABv6, którą w uproszczeniu, można traktować jako część sieci PL-LAB. Różnica pomiędzy sieciami tj. PL-LABv6 i PL-LAB jest jednak taka, że sieć PL-LAB zestawia połączenia na poziomie MPLS i Ethernetu, podczas gdy sieć PL-LABv6 jest rozszerzona o warstwę wyższą, oferując łączność na poziomie protokołu IPv6 z dodatkowymi usługami sieciowymi niezbędnymi dla działania tego protokołu. Reasumując, sieć PL-LABv6 wykorzystuje łączność na poziomie Ethernet zestawianą poprzez sieć PL-LAB dodając do niej dodatkowe funkcjonalności przeznaczone dla realizacji łączności z wykorzystaniem protokołu IPv6.

Sieć PL-LABv6 jest siecią ogólnopolską, łączącą laboratoria (wszystkich zespołów projektu uczestnicących w pracach projektu - Internet IPv6) z wykorzystaniem protokołu routingu BGP i dedykowanych do tego połączeń w ramach jednej puli adresowej przedstawionej w tabeli 1.

**Tablica 1.** Przydział adresu IPv6 dla każdego z laboratoriów

| Laboratorium | Adres IPv6 |
|---|---|
| IŁ | 2001:808:2:31ff::3/64 |
| PCSS | 2001:808:2:31ff::4/64 |
| PG | 2001:808:2:31ff::5/64 |
| PW | 2001:808:2:31ff::8/64 |
| PWr | 2001:808:2:31ff::9/64 |

Każde z laboratoriów posiada własny router BGP wraz z numerem AS i osobną pulą adresową używaną w ramach każdego z laboratoriów. Takie podejście umożliwia rozproszenie konfiguracji IP i uelastycznienie laboratoriów względem sposobu adresacji sieci wewnętrznej, wykorzystywanej puli adresowej, numeru AS itp. Ponadto, w przypadku podłączenia sieci PL-LABv6 do Internetu światowego, łączność do każdego z laboratoriów może odbywać się poprzez lokalnego dostawcę Internetu (którym jest sieć akademicka MAN), co zdaje się być rozwiązaniem najbardziej optymalnym.

Zasoby sprzętowo-sieciowe w każdym z laboratoriów, w miastach uczestników, różnią się nieco między sobą, co wynika z różnorodności prowadzonych prac. W tabeli 2 przedstawiono zestawienie poszczególnych zasobów sprzętowych.

## 3 Eksperymenty

Jednym z istotnych celów prac realizowanych w sieci PL-LABv6 jest przeprowadzanie eksperymentów z wykorzystaniem protokołu IPv6. Eksperymenty te dotyczą zarówno weryfikacji poprawności działania protokołów (takich jak np.

**Tablica 2.** Zestawienie zasobów sprzętowych i usług każdego z laboratoriów

| Lokalizacja | Sprzęt | Usługi |
|---|---|---|
| Poznań – laboratorium PCSS | – 1 x router MX240 Juniper MX router<br>– 1 x przełącznik EX3200 Juniper<br>– 1 x serwer HP | Na poziomie produkcyjnym:<br>– DNS<br>– DHCPv6<br>Eksperymentalnie i na potrzeby badawcze:<br>– DHCPv6 dibbler<br>– IPTV<br>– klient, serwer VoIP<br>– SNMP manager, SNMP agent<br>– Tcp dump tool – narzędzie do zbierania informacji o ruchu wewnątrz sieci i udostępniania ich poprzez protokół SNMP<br>– pmacct, pmacct SNMP |
| Warszawa – laboratorium IŁ | – 1 x router MX240 Juniper MX router<br>– 1 x przełącznik EX3200 Juniper<br>– 2 x serwer HP<br>– 1 x punkt dostepowy WiFi<br>– 1 x sprzętowy klient protokołu VoIP | Na poziomie produkcyjnym:<br>– DNS<br>– DHCPv6<br>Eksperymentalnie i na potrzeby badawcze:<br>– DHCPv6 dibbler<br>– IPTV na platformie IMS |
| Warszawa – Laboratorium PW | – 1 x router MX240 Juniper MX router<br>– 1 x przełącznik EX3200 Juniper<br>– 1 x serwer (Intel Core i5-750 processor, 4GB RAM, 500GB HDD),<br>– 1 x punkt dostepowy WiFi (N600 Wireless Dual Band Gigabit Router WNDR3700 with DLNA support) | Na poziomie produkcyjnym:<br>– DNS<br>– DHCPv6 |
| Wrocław – laboratorium PWr | – 1 x router MX240 Juniper MX router<br>– 1 x przełącznik EX3200 Juniper<br>– 2 x serwer HP | Na poziomie produkcyjnym:<br>– DNS<br>– DHCPv6<br>Eksperymentalnie i na potrzeby badawcze:<br>– DHCPv6 dibbler |
| Gdańsk – laboratorium PG | – 1 x router MX240 Juniper MX router<br>– 1 x przełącznik EX3200 Juniper<br>– 3 x serwer HP | Na poziomie produkcyjnym:<br>– DNS<br>– DHCPv6<br>Eksperymentalnie i na potrzeby badawcze:<br>– DHCPv6 dibbler<br>– Niezawodna telekonferencja VoIP IPv6 na platformie IMS<br>– Proxy mobile IPv6 |

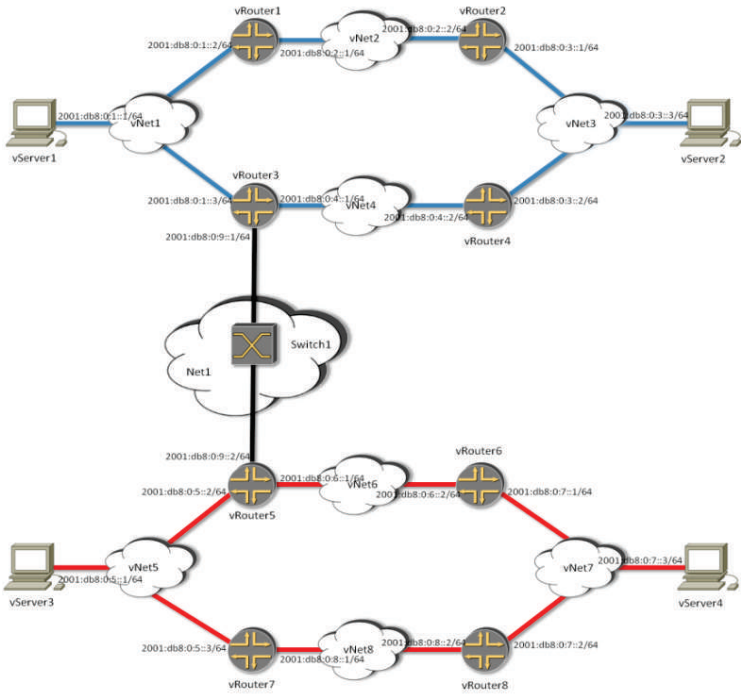
DHCPv6, DNS, BGP) dla różnych scenariuszów wdrożeń jak również testowania nowych narzedzi i aplikacji rozwijanych w ramach samego projektu IIP (jak np. Dibbler, pmacct, iPTV, VoIP, LiveCD Test Tool, Przewodnik Migracji do IPv6, ip46net, niezwodna telekonferencja VoIP na platformie IMS, Proxy Mobile IPv6). Poniżej przedstawiono krótkie charakterystyki realizowanych eksperymentów związanych z tworzonymi w CS1 narzedziami i aplikacjami , które można wykorzystać do innych badań i eksperymentów realizowanych poza projektem IIP.

### 3.1 Systemy wirtualizacji

W projekcie IIP uruchomiono i badano różnorodne systemy wirtualizacji takie jak Cytrix XEN server, Microsoft Hyper-V server, VMware ESXi. W ramach uruchomionych wystemów wirtualnych testowano poprawność działania przepływności ruchu sieciowego (porównywano system wirtualny z systemem fizycznym), działanie protokołów routingu OSPF, BGP (z wykorzystaniem routerów programowych Vyatta jak również routerów wirtualnych działających na sprzęcie Juniper MX240) [2], działanie usług sieciowych DNS i DHCPv6 oraz weryfikowano stopień obsługi IPv6 dla systemów zarządzania wirtualizatorami wraz z wykorzystaniem protokołu SNMP oraz innych protokołów takich jak IPsec. Na rysunkach rys. 1, rys. 2, przedstawiono różne topologie sieciowe zbudowane z wykorzystaniem środowisk wirtualnych, których używano do przeprowadzania eksperymentów. Dzięki wykorzystaniu systemów wirtualizacji możliwe jest budowanie skomplikowanych topologii sieciowych z wykorzystaniem routerów wirtualnych na platformie Juniper MX240, jak również tworzenie wielu zwirtualizowanych systemów operacyjnych uruchamianych w tych topologiach. Możliwe jest uruchamianie dowolnych systemów operacyjnych (które mogą być nawet dostarczone przez użytkownika-badacza), pod warunkiem że są wspierane przez wirtualizatory, wraz z zainstalowanymi aplikacjami i usługami sieciowymi. Usługi te i aplikacje mogą być poddane eksperymentom lub do nich wykorzystywane.

### 3.2 Testy protokołów routingu

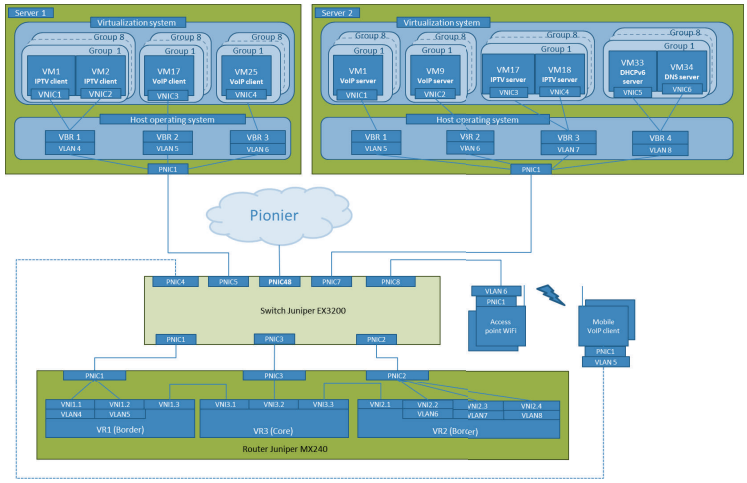
Dzięki systemom wirtualizacji i elastyczności budowania topologii sieciowych przedstawionych w punkcie 3.2 możliwe jest przeprowadzanie testów protokołów routingu dla różnych konfiguracji sieci. W trakcie przeprowadzanych prac realizowano testy dla protokołów OSPFv6 dla IPv6 dla topologii przedstawionej na rys. 1, gdzie sprawdzano poprawność wymiany informacji o sieciach w obszarze działania protokołu oraz sprawdzano wykorzystanie tras zapasowych, umiejętność szybkiego wykrycia awarii i przełączenie ruchu. W przypadku protokołu BGP sprawdzano poprawność wymiany informacji o sieciach dla różnych systemów autonomicznych. Protokoły te mogą zostać wykorzystane do eksperymentów wykorzystywanych do badań poza projektem. Możliwe jest również, w razie konieczności, uruchamianie innych protokołów routingu, o ile są wspierane dla posiadanych zasobów sprzętowo-programowych.




**Rysunek 1.** Przykładowa topologia sieciowa laboratorium z wykorzystaniem routerów programowych i zwirtualizowanych systemów operacyjnych.

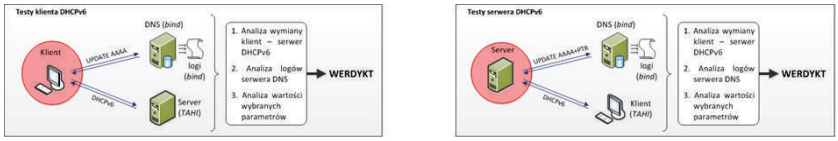
### 3.3 Testy usług i aplikacji unikalnych dla IPv6

Protokół IPv6 wymaga, podobnie jak jego poprzednik IPv4, określonych usług sieciowych dla zapewnienia pełnej funkcjonalności. Usługami tymi są w szczególności DHCPv6 oraz DNS. W ramach projektu IIP opracowano podstawową wersję konfiguracji dla tych usług. Przetestowano poprawność ich działania w środowiskach zwirtualizowanych przedstawionych na rys. 1 i 2. W ramach IIP rozwijane są również implementacje klienta i serwera DHCPv6, pod nazwą Dibbler. Dibbler jest otwartym oprogramowaniem implementowanym na poziomie produkcyjnym dla wielu systemów operacyjnych oraz platform sprzętowych [3]. Prowadzone prace implementacyjne umożliwiają wdrażanie dodatkowych funkcjonalności do protokołu DHCPv6 opisanych w specyfikacjach draft takich jak: RFC6334: DS-Lite tunnels configuration, draft-ietf-dhc-dhcpv6-redundancy-consider-02, draft-ietf-mif-dhcpv6-route-option-03, draft-ietf-dhc-dhcpv6-failover-requirements-00, draft-mrugalski-dhc-dhcpv6-suboptions, draft-mdt-softwire-map-dhcp-option-01. Jeden z nurtów prac poświęcony jest testom zgodności serwera i klienta DHCPv6 w zakresie funkcjonalności niedostępnych jeszcze dla współczesnych testerów, obejmujących m. in. specyfikacje: The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name




**Rysunek 2.** Przykładowa topologia sieciowa laboratorium z wykorzystaniem routerów wirtualnych na platformie Juniper MX240 i zwirtualizowanych systemów operacyjnych.

(FQDN) Option - RFC 4704, Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6) - RFC 3898 (por. rys. 3).




**Rysunek 3.** Testy zgodności DHCPv6.

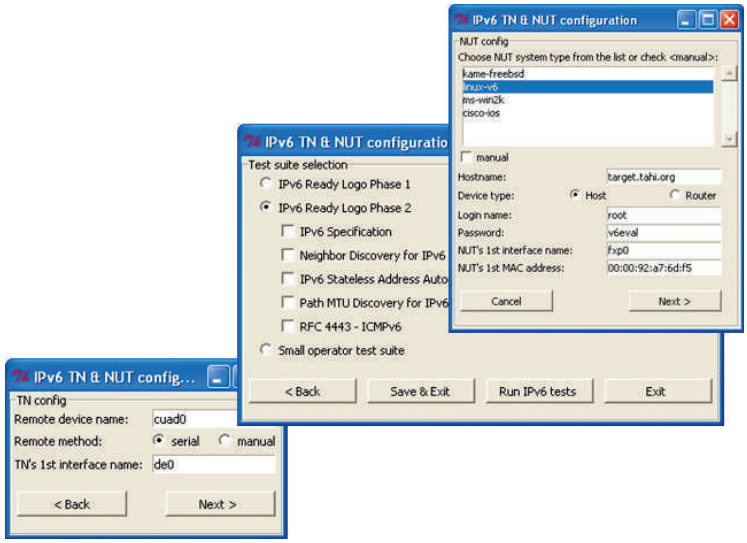
Wśród aplikacji dedykowanych dla środowiska IPv6, realizowano prace związane z implementacją systemu telewizji internetowej (IPTV) oraz niezawodnej telekonferencji VoIP na platformie IMS. Zaimplementowane algorytmy obejmują równoważenie obciążenia ruchu alokujące konferencje i ich zarządzanie na dostępne w danej chwili serwery mediów.

W ramach prac dotyczących mobilności użytkowników w sieciach IPv6 opracowano jedną z pierwszych na świecie implementacji protokołu Proxy Mobile IPv6. Protokół Proxy Mobile IPv6 jest rozwiązaniem pozwalającym na realizację mobilności bez konieczności modyfikacji stosu IPv6 w terminalu klienta (tzw. Network-based mobility management). W zaproponowanym podejściu cała funkcjonalność mobilności realizowana jest w rdzeniu sieci. Pozwala to na lepsze wykorzystanie pasma w bezprzewodowej sieci dostępowej. W porównaniu do trady-

cyjnego rozwiązania Mobile IP w tym przypadku nie jest stosowane tunelowanie na fragmencie łącza dostępowego. W efekcie nie występuje narzut protokolarny obniżający przepływność dostępną dla użytkownika. Ponieważ cała funkcjonalność odbywa się w rdzeniu sieci operator ma większą możliwość sterowania oraz kontroli procesu przełączania. Poszczególne komponenty sieci wspierające mobilność: LMA (Local Mobility Anchor) oraz MAG (Media Access Gateway) mogą być implementowane w róznych lokalizacjach [4].

### 3.4 Zasady współpracy IPv6/IPv4, testowanie poprawności wdrożeń oraz zagadnienia migracji

Opracowano szereg aplikacji i rozwiązań, które mogą być pomocne do różnorodnych badań oraz wdrożeń protokołu IPv6. Jedną z aplikacji jest LiveCD Test Tool rys 4, która udostępniana wraz z samodzielnym systemem operacyjnym jest gotowym natychmiast do użycia narzędziem testowym. Umożliwia ona przeprowadzenie różnorodnych ustandaryzowanych testów referencyjnych w oparciu o platformę TAHI [5].




**Rysunek 4.** LiveCD test tool.

Ponadto zaimplementowano narzędzie tłumaczące IPv4 na IPv6 ip46nat, którego celem jest testowanie i porównywanie różnorodności rozwiązań w kierunku opracowania uniwersalnego translatora miedzy protokołami IPv4 i IPv6.

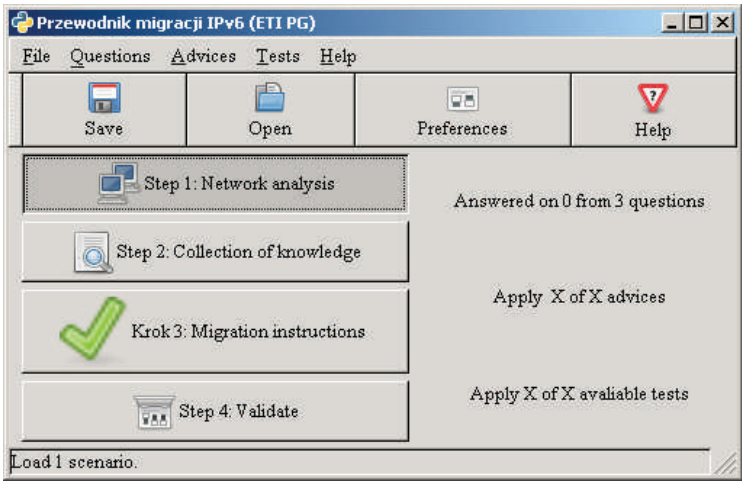
Kolejną aplikacją, bardzo przydatną na etapie wdrażania protokołu IPv6 jest Przewodnik migracji do IPv6 [6][7][8]. Zadaniem tej aplikacji jest pomoc

dla administratorów sieci i użytkowników sieci SOHO w procesie migracji do protokołu IPv6.

Aplikacja Przewodnika migracji do IPv6 jest interaktywnym narzędziem zawierającym bazę wiedzy obejmującą porady i wiedzę dotyczącą procesu migracji do IPv6 (patrz rys. 5) [9][10]. Automatyczne skanowanie zasobów sieci oraz współpraca aplikacji z użytkownikiem pozwala na dostarczenie użytkownikowi gotowych porad dotyczących migracji systemów operacyjnych, usług sieciowych oraz aplikacji do pracy z protokołem IPv6. Baza wiedzy Przewodnika migracji oprócz porad dotyczących konfiguracji konkretnych usług i aplikacji w większości powszechnie stosowanych sieciowych systemów operacyjnych obejmuje również szeroki zakres specjalistycznej wiedzy dotyczącej protokołu IPv6.

Integracja Przewodnika migracji do IPv6 z zestawem wcześniej omówionych testów w postaci płyty LiveCD umożliwia wykonanie kompleksowej analizy istniejącej konfiguracji sieci, dostarczenie fachowej wiedzy dotyczącej procesu migracji oraz wykonanie testów weryfikujących poprawność przeprowadzenia migracji. Zintegrowane narzędzie w postaci LiveCD może być niezwykle użyteczne zarówno dla administratorów sieci jak i dla zwykłych użytkowników systemów komputerowych dostarczając wiedzy i porad dotyczących użytkowanych systemów.

W oparciu o wiedzę i porady zawarte w Przewodniku migracji możliwe jest wytworzenie narzędzi wspomagających oraz wykonujących migrację usług i aplikacji sieciowych. W zakresie prac prowadzonych w ramach grantu badawczego zrealizowano sytemy automatycznie migrujące aplikacje Apache HTTP Server i ProFTPD Server oraz konfiguracje usług DNS i DHCP i routingu dynamicznego na routerach Cisco, Vyatta i OpenWRT.




**Rysunek 5.** Przewodnik migracji IPv6.

## 4 Podsumowanie

Zaprezentowana w artykule ogólnopolska sieć PL-LABv6, rozumiana jako krajowe środowisko testowe protokołu IPv6 wraz z szeregiem usług, aplikacji i scenariuszy jest oferowana dla polskiego środowiska badawczego jako pomoc przy wdrażaniu protokołu IPv6, tworzeniu scenariuszy oraz przeprowadzaniu innych prac badawczych. Zaprezentowane konfiguracje mogą być uruchamiane w różnych wariantach w zależności od potrzeb badaczy, począwszy od eksperymentów, które mogą być prowadzone w oparciu o laboratorium lokalne w jednym z miast, a skończywszy na bardziej zaawansowanych topologiach sieciowych, obejmujących sieć ogólnopolską. Dzięki systemom wirtualizacji, możliwe jest zrównoleglenie prac, izolacja środowisk, jak również lepsze wykorzystanie sprzętu oraz implementacja bardziej zaawansowanych topologii.

## Literatura

1. Projekt Inżynieria Internetu Przyszłości, `https://www.iip.net.pl`
2. PL-LAB: sieć eksperymentalna projektu „Inżynieria Internetu Przyszłości, Krajowe Sympozjum Telekomunikacji i Teleinformatyki KSTiT 2011, konferencja KST-IT 2011.
3. H. Gut, J. Mongay Batalla, W. Latoszek, B. Gajda, W. Procyk, K. Chudzik, J. Kwiatkowski, K. Nowak: Testowanie usług IPv6 w sieci testowej opartej na urządzeniach z możliwością wirtualizacji sprzętowe", Krajowe Sympozjum Telekomunikacji i Teleinformatyki KSTiT 2011,
4. M. Hoeft, K. Gierłowski, T. Gierszewski, J. Konorski, K. Nowicki, J. Woźniak: Measurements of QoS/QoE Parameters for Media Streaming in a PMIPv6 Testbed with 802.11 b/g/n WLAMs, Metrol. Meas. Syst., Vol. XIX (2012), No2, pp. 283-294, ISSN 0860-8229
5. TAHI project: `http://www.tahi.org`
6. W. Gumiński, A. Mrugalska, M. Stankiewicz, K. Nowicki, J. Woźniak Przewodnik migracji do IPv6, Przegląd telekomunikacyjny i wiadomości telekomunikacyjne, 8-9/2011, s. 1184-1192
7. W. Gumiński, J. Światowiak, Obsługa protokołu IPv6 w systemach operacyjnych Windows i Linux, Przegląd Telekomunikacyjny i wiadomości telekomunikacyjne, 8-9/2010, s. 797-798
8. K. Podolski, K. Nowicki: The IPv4 to IPv6 MIGRATION Of APPLICATIONS, ICT Young 2012, Gdańsk, 26,27 maj 2012
9. K. Nowicki, M. Stankiewicz, A. Mrugalska, T. Mrugalski, J. Woźniak, Knowledge Managemenent in the IPv6 Migration Process, Computer Networks 2011, Czerwiec 2011, s. 114-124
10. K. Nowicki, M. Stankiewicz, A. Mrugalska, J. Woźniak, T. Mrugalski, Extension management of a knowledge base migration process to IPv6, The 11th IEEE IPSJ International Symposium on Applications and the Internet, Munich, July 2011, s. 497-501

# Wykaz autorów


Adamczyk, Błażej, 25

Belter, Bartosz, 165
Bęben, Andrzej, 1, 135, 146, 165, 179
Białoń, Paweł, 165
Bielecki, Jakub, 38
Binczewski, Artur, 308
Borzdyński, Jakub, 146
Brachman, Agnieszka, 105
Brzostowski, Krzysztof, 87
Burakowski, Wojciech, 1, 119

Cabaj, Krzysztof, 46
Cholewa, Michał, 179
Chudzik, Krzysztof, 279, 308
Chydziński, Andrzej, 25

Dolata, Łukasz, 194
Dombek, Krzysztof, 11
Drapała, Jarosław, 87

Gajda, Bartosz, 279, 308
Gierłowski, Krzysztof, 105, 220
Gierszewski, Tomasz, 220
Głomb, Przemysław, 179
Gozdecki, Janusz, 38, 77, 87, 119
Granat, Janusz, 1, 38, 165
Grzech, Adam, 119
Gumiński, Wojciech, 262, 308
Gut, Henryk, 308
Gutkowski, Jakub, 146, 165

Hanczewski, Sławomir, 105
Hoeft, Michał, 220

Juszczyk, Artur, 11
Juszczyszyn, Krzysztof, 87

Kaliszan, Adam, 105
Kamiński, Paweł, 262
Kamisiński, Andrzej, 46
Kantor, Mirosław, 38, 77
Kasperek, Jerzy, 46
Kołaczek, Grzegorz, 46
Konorski, Jerzy, 46
Kosek-Szott, Katarzyna, 119
Kosiedowski, Michał, 179
Kotulski, Zbigniew, 46
Kozik, Andrzej, 146
Krawiec, Piotr, 146, 165, 179, 194, 209
Krzywania, Radosław, 194
Kuśmierek, Ewa, 179
Kwiatkowski, Jan, 279, 308

Latoszek, Waldemar, 308
Lewandowski, Bartosz, 87, 119

Łopatowski, Łukasz, 165
Łoska, Łukasz, 262
Łoziak, Krzysztof, 119

Mazurek, Cezary, 87, 119
Mongay Batalla, Jordi, 146, 209, 308
Mrugalski, Tomasz, 293, 308

Natkaniec, Marek, 87, 119
Nowak, Kamil, 279, 308
Nowicki, Krzysztof, 262, 293, 308

Ogrodowczyk, Łukasz, 105
Olender, Paweł, 165
Olszewski, Iwo, 11, 105
Opozda, Sebastian, 179
Ostapowicz, Piotr, 146
Ożga, Mateusz, 293

Pacyna, Piotr, 46
Parniewicz, Damian, 11
Podolski, Karol, 262
Procyk, Wiktor, 279, 308
Przywecki, Miłosz, 194

Radziuk, Arkadiusz, 87, 119
Rajda, Paweł, 46
Rak, Jacek, 38, 77
Rawski, Mariusz, 105
Rogowski, Jan, 87
Romaszewski, Michał, 179
Romaszkan, Wojciech, 46
Rudek, Radosław, 146
Rupiński, Marcin, 46
Rzepka, Dominik, 46

Sikora, Marek, 87

Skałecki, Piotr, 279, 308
Sochan, Arkadiusz, 179
Stroiński, Aleksander, 179
Szałachowski, Paweł, 46
Szott, Szymon, 119
Szymak, Wojciech, 38

Świątek, Paweł, 87, 119

Tarasiuk, Halina, 1, 87, 105, 119

Wajda, Krzysztof, 38, 77
Walczak, Dariusz, 87, 119
Wasilewski, Krzysztof, 262
Wiśniewski, Piotr, 11, 105, 146, 165
Woźniak, Józef, 220, 308
Wrzos, Marcin, 87

Zimnowoda, Jehoszafat, 46