Koncepcja systemu multikastowej aktualizacji zasobów serwerów korzystających z adresacji anykastowej

Robert R. Chodorek, Agnieszka Chodorek, Andrzej R. Pach (e-mail: chodorek@kt.agh.edu.pl, a.chodorek@tu.kielce.pl, pach@kt.agh.edu.pl)
Katedra Telekomunikacji Akademii Górniczo-Hutniczej, Kraków,
Samodzielny Zakład Telekomunikacji i Fotoniki, Politechnika Świętokrzyska, *Kielce*

STRESZCZENIE

Anycast jest nowym schematem adresacji, dostarczającym usługi transmisyjnej typu 1-do-(1zN). Anycast jest adresacją zorientowaną na usługę – każda usługa ma swój predefiniowany adres IP, identycznych dla wszystkich serwerów, które świadczą tę usługę, niezależnie od ich lokalizacji geograficznej. Wybór serwera, który będzie świadczył usługę danemu użytkownikowi (najlepiej: najbliższego), następuje w sposób automatyczny. Aby usługa korzystająca z adresacji anycast mogła efektywnie funkcjonować, dane na wszystkich serwerach informacyjnych, z którymi może podłączyć się użytkownik muszą być spójne i muszą być aktualizowane jednocześnie. W artykule zaprezentowania zostanie koncepcja efektywnej aktualizacji zawartości serwerów, zrealizowana na bazie transmisji multikastowej.

ABSTRACT

A concept of multicast updating of content of servers, which utilize anycast addressing

Anycast is a new addressing scheme, which provides a one-to-one-of-many communication service. Anycast is a service-oriented addressing – each service has its own, pre-defined IP address, identical for all servers, which provides the service, apart from their geographical location. The IP packet is delivered to the one of servers with the same anycast address (preferably the closest one) and the server selection take place automatically. The anycast-based service will work effectively only if data stored in all servers are coherent and if they are updated simultaneously. In the paper, a concept of multicast updating of content of servers which utilize anycast addressing is presented.

1. Wprowadzenie

Wraz z rosnącą komercjalizacją i gwałtownym rozwojem sieci Internet nieustannie wzrasta zapotrzebowanie na dostępne zasoby sieciowe. Aby sprostać tym wymaganiom, budowane są sieci o coraz to wyższych przepustowościach oraz opracowuje się coraz to wydajniejsze protokoły. Uzupełnieniem tych działań jest optymalizacja dostępu do zasobu, tworzona na bazie dodatkowej infrastruktury. Od dawna do dystrybucji stron WWW stosowane są metody, które pozwalają na przyspieszenie dostępu do informacji i ograniczenie ruchu w sieci do niezbędnego minimum. Do metod tych należą m.in. serwery proxy, pierwotnie opracowane w celu umożliwienia dostępu do serwisów WWW klientom korzystającym z zabezpieczeń typu firewall [13]. Serwery te nie tylko pośredniczyły w przekazywaniu stron (realizując niekiedy filtrację przekazywanych informacji), ale, niejako dodatkowo, pracowały jako pamięć podręczna (*cache*). Gromadziły zawartość stron WWW, w których przekazywaniu pośredniczyły, co w przypadku kolejnego odwołania się do danej strony, pozwalało na szybsze i efektywniejsze przesłanie strony od razu z serwera proxy. Dzięki temu spadało także obciążenie serwerów WWW. Z czasem ta dodatkowa funkcja serwera proxy stała się jego funkcją „sztandarową".

Aktualizacja serwerów proxy nie jest sprawą prostą. W tym celu stosowane są specjalizowane rozwiązania (jak chociażby protokół ICP, *Internet Cache Protocol* [21]), które często organizowane są hierarchicznie [20]. Inną metodą optymalizacji dostępu do zasobu jest powielanie zasobów (*mirroring*). Ze względu na uzyskiwaną w ten sposób redundancję, metoda ta stosowana jest często w celu zwiększania niezawodności typowych usług sieciowych, jak chociażby DHCP (*Dynamic Host Configuration Protocol*) czy NHRP (*Next Hop Resolution Protocol*), niejako „przy okazji" dając lepszy dostęp do zasobów (uzyskiwany dzięki odpowiedniej lokalizacji serwerów). Powielanie zasobów w celu poprawy ich dostępności spotykane jest często w przypadku dystrybucji powszechnie wykorzystywanych zasobów (np. dokumentów RFC (*Request For Comments*), czy dystrybucji systemu Linux).

W przypadku usług typu DHCP czy NHRP, ilość gromadzonych informacji bywa stosunkowo niewielka, jednakże serwery tych usług muszą zawsze posiadać identyczne, spójne dane. Aktualizacja serwerów usług musi być zorientowana na spełnienie tych wymagań. Przykładowo, w sieciach NBMA (*Non Broadcast Multiple Access*) jest ona realizowana z wykorzystaniem protokołu SCSP (*Server Cache Synchronization Protocol*) [12]. W przypadku zasobów typu dokumenty RFC czy dystrybucje systemu Linux, ilość gromadzonych informacji bywa duża, jednak dopuszczalne jest chwilowe (zwykle nieprzekraczające jednego dnia) opóźnienie w aktua-

lizacji serwerów. Aktualizacja takich serwerów odbywa się z użyciem odpowiednich protokołów warstwy aplikacji, korzystających (w warstwie transportowej) z protokołu TCP (*Transmission Control Protocol*).

Istnieje jednak pewna klasa usług, które charakteryzują się potencjalnie dużą ilością gromadzonych informacji i wymagającymi jednoczesnej (niejednokrotnie z dokładnością do opóźnień sieci – np. notowania giełdowe) aktualizacji serwerów. W artykule zostanie zaproponowana koncepcja serwisu świadczącego taką usługę. Serwis ten wykorzystuje adresację anykastową do lokalizacji optymalnego (z punktu widzenia efektywności dostępu do usługi) serwera oraz transmisję multikastową do jednoczesnej aktualizacji zawartości serwerów. Proponowana koncepcja zakłada zbudowanie sieci nakładkowej (*overlay network*), łączącej serwery dedykowane dla usługi, które mogą być (jeżeli istnieje taka potrzeba i możliwości techniczne) dodatkowo uzupełnione serwerami typu proxy.

Potencjalne zastosowania serwisów zbudowanych na podstawie proponowanej koncepcji są bardzo szerokie – od dystrybucji informacji serwisów WWW, poprzez dystrybucję baz danych, na dystrybucji informacji multimedialnej kończąc. Pomimo że system może być stosowany do dystrybucji stron WWW, nie należy go bynajmniej rozpatrywać jako alternatywy dla usług typu proxy. Proponowana koncepcja różni się od serwerów proxy dystrybucją pełnej zawartości konkretnego serwisu WWW (a nie tylko żądanych ostatnio stron WWW), przy czym klient (grupa klientów) może korzystać z pośrednictwa serwera proxy wspierającego adresację anykastową.

Niniejszy artykuł ma na celu zaproponowanie koncepcji multikastowej aktualizacji zawartości serwerów korzystających z adresacji anykastowej. Rozdział 2 zawiera opis elementów proponowanego systemu aktualizacji serwerów usług. W rozdziale 3 zostanie przedstawiony opis działania proponowanego systemu dystrybucyjnego. Rozdział 4 omawia operacje rejestracji i wyrejestrowywania serwera usług w serwerze matce. Rozdział 5 poświęcony został w całości problematyce dystrybucji danych pierwotnych. Rozdział 6 stanowi podsumowanie niniejszego artykułu.

2. Elementy systemu aktualizacji serwerów usług

Anykast jest schematem adresacji IP, dostarczającym usługi transmisyjnej typu 1-do-(1zN). Adresacja typu anykast została wprowadzona do protokołu IP w wersji 6 jako odpowiedź na rosnące zapotrzebowanie na tego typu usługę ze strony szybko rozwijającego się rynku usług multimedialnych [4]. Prowadzono również próby wprowadzenia transmisji anykast do wersji 4 protokołu IP [17], jednak nigdy nie wyszły one ze stadium eksperymentu.

W sieciach IP, transmisja anykast może być wykorzystywana w przypadku współpracy klienta z grupą serwerów udostępniających zasoby wielu klientom, przy czym każdy z serwerów musi wówczas posiadać identyczny (z punktu widzenia klienta) zbiór zasobów (*mirror*). Klient może połączyć się z dowolnym serwerem usług, podając adres anykast grupy serwerów. Zostanie wówczas połączony z najbliższym serwerem usług, przy czym miarą „bliskości" serwera jest metryka danej trasy, znajdująca się w tablicy rutingu. W efekcie, użytkownik nie musi znać unikalnego adresu IP najbliższego (w sensie metryki) serwera, aby korzystać z danej usługi. Potrzeba i wystarczy, by znał jedynie adres anykastowy tej usługi.

Aby usługa korzystająca z adresacji anykastowej mogła efektywnie funkcjonować, dane na wszystkich serwerach informacyjnych, z którymi może połączyć się użytkownik, muszą być spójne i muszą być aktualizowane jednocześnie. Zbudowanie takiej efektywnej aktualizacji jest możliwe na bazie niezawodnej transmisji multikastowej pomiędzy wszystkimi serwerami.

Proponowana przez Autorów koncepcja systemu aktualizacji zasobów serwerów usług korzystających z adresacji anykastowej zakłada współdziałanie czterech elementów systemu (rys. 1):

1) serwera matki, na którym znajdują się dane pierwotne, podlegające dystrybucji do serwerów usług;
2) serwerów usług, na których znajdują się kopie danych pierwotnych z serwera matki, podlegające redystrybucji do stacji klienckich;
3) stacji klienckich, pobierających dane z serwerów usług;
4) infrastruktury sieciowej IPv6 łączącej te wszystkie elementy.

Serwer matka pełni podwójną rolę: dostawcy treści przekazu oraz rolę systemu koordynującego. Jako dostawca treści przekazu, serwer matka gromadzi dane pierwotne, które zawierają treści specyficzne dla danej usługi. Mogą to być:

- materiały filmowe – np. w przypadku usług typu telewizja internetowa, czy media na żądanie (jak chociażby wideo na żądanie, ale również informacje na żądanie);
- dźwięk – np. w przypadku usług typu radio internetowe, czy informacje na żądanie;
- dane – np. pliki (w tym dystrybucje oprogramowania, uaktualnienia oprogramowania), ale również dane giełdowe, telemetryczne itp., strony WWW, zawartość różnego typu baz danych.

Administrator serwera matki rejestruje globalny adres anykastowy świadczonej usługi, jak również rejestruje globalny adres multikastowy, używany do dystrybucji danych pierwotnych.

Jako system koordynujący pracę usługi, serwer matka:

- rejestruje serwery usług;
- jeżeli usługa tego wymaga, dokonuje autoryzacji serwerów usług;

- przekazuje zarejestrowanym serwerom usług adres anykastowy świadczonej usługi;
- przekazuje zarejestrowanym serwerom usług adres multikastowy, używany do dystrybucji danych pierwotnych;
- jeżeli usługa tego wymaga, tworzy i rozsyła klucze do szyfrowania dystrybucji danych pierwotnych;
- dokonuje aktualizacji danych pierwotnych;
- dokonuje chwilowego bądź stałego wyrejestrowania serwera usług.

Aktualizacja danych pierwotnych – „sztandarowa" funkcja serwera matki – jest realizowana metodą wypychania (*push*). Serwer matka „wypycha" przygotowane uprzednio dane, zmuszając wszystkie aktywne serwery usług do równoczesnej aktualizacji zawartości.

Serwer usług jest dostępny dla użytkownika (stacji klienckiej) poprzez swój adres anykastowy. Serwer ten przechowuje lokalną kopię danych pierwotnych, otrzymaną z serwera matki oraz świadczy lokalne usługi dystrybucyjne (w zależności od charakteru usługi, unikastowo lub multikastowo). W szczególności, zadania serwera usług obejmują:

- rejestrację klientów;
- jeśli usługa tego wymaga, również autoryzację klientów i dystrybucję kluczy;
- dostarczanie klientowi adresu usługi dystrybucyjnej (w zależności od usługi: adresu unikastowego lub adresu multikastowego);
- redystrybucję (w trybie unikast lub multikast) całości lub fragmentów lokalnej kopii danych pierwotnych;
- jeżeli usługa tego wymaga, rozliczenia z użytkownikami i z dostawcą danych pierwotnych;
- wyrejestrowywanie klientów.

Serwer matka oraz serwery usług mogą należeć do tego samego dostawcy, jak również do różnych dostawców. W szczególności, serwer matka oraz każdy z serwerów usług mogą należeć do innego właściciela.

Stacja kliencka pozyskuje dane z serwera usług. W zależności od charakteru usługi, może być to realizowane unikastowo lub multikastowo. Jeżeli redystrybucja danych realizowana jest w trybie unikast, to pozyskiwanie danych odbywa się zgodnie z metodą „wyciągania" (*pull*). Stacja kliencka pozyskuje od serwera usług tylko te treści, którymi jest zainteresowany użytkownik, a pozyskiwanie danych odbywa się na zlecenie stacji. Jeżeli redystrybucja danych realizowana jest w trybie multikast, to odbywa się to metodą wypychania (*push*). Serwer usług „wypycha" dane przygotowane do wysłania, a wszystkie aktywne stacje klienckie, dołączone do tego serwera, odbierają te dane. W przypadku danych giełdowych oraz audycji telewizyjnych i radiowych pożądane jest, by wszyscy klienci odbierali dane równocześnie. Serwer usług spełnia wówczas rolę serwera proxy, który odbiera, w razie potrzeby przekodowuje i (lub) filtruje dane, a następnie przekazuje je dalej. Ewentualne przechowywanie danych jest realizowane tylko na potrzeby redystrybucji.

Oprócz pozyskiwania danych, stacja kliencka dokonuje również:

- pozyskania informacji o usłudze;
- rejestracji w serwerze usług, osiąganym dzięki adresacji anykastowej;
- jeżeli usługa tego wymaga, pozyskania kluczy deszyfrujących przekaz;
- jeżeli usługa tego wymaga, sterowania przebiegiem sesji (np. przewijanie materiału filmowego w przypadku usługi wideo na żądanie);
- wyrejestrowania z serwera usług.

Stacja kliencka może być realizowana w oparciu o komputer klasy PC, laptop lub dowolne inne urządzenie końcowe, np. urządzenia abonenckie telefonii komórkowej UMTS (*Universal Mobile Telecommunications System*). Jeżeli charakter usługi tego wymaga, stacją kliencką może być specjalizowane urządzenie (np. *set-top-box* przeznaczony do odbioru telewizji internetowej).

Infrastruktura sieciowa łączy ze sobą elementy systemu: zarówno serwer matkę z serwerami usług, jak i stacje klienckie z optymalnym (w sensie zadanej metryki) serwerem usług. Pełnienie tej funkcji wymaga zarówno zastosowania typowych elementów infrastruktury sieciowej IPv6, jak rutery IPv6, łącza, protokoły, jak i występowania specjalizowanych rozwiązań do przydziału i koordynacji adresacji multikastowej i anykastowej. Dodatkowo, w sieci IPv6 wymagany jest specjalizowany ruting anykastowy. Jeżeli warstwa transportowa jest realizowana z wykorzystaniem protokołu PGM (*Pragmatic General Multicast*), aby system był w pełni skalowany, również rutery (a co najmniej: wybrane rutery) muszą wspierać ten protokół. Podobna sytuacja występuje w przypadku protokołu NORM, który również do efektywnej pracy potrzebuje wsparcia ruterów.

3. Opis działania systemu dystrybucyjnego

Serwer matka, jeden lub kilka serwerów usług, jedna lub kilka stacji klienckich oraz łącząca je infrastruktura sieciowa IPv6 tworzą razem pełny system dystrybucyjny świadczonej usługi. Działanie tego systemu można podzielić na dwa etapy.

Etap I związany jest z aktualizacją zasobów na serwerach usług. Obejmuje on procedury:

- związane z rejestracją i wyrejestrowywaniem się serwera usług w serwerze matce,
- związane z dystrybucją danych pierwotnych do serwerów usług.

Procedury związane z rejestracją realizowane są na zlecenie serwera usług. Zostały one dokładniej omówione w rozdziale 4. Procedury związane z dystrybucją

danych pierwotnych są realizowane na zlecenie serwera matki. Zostały one dokładniej opisane w rozdziale 5.

Etap II związany jest z redystrybucją kopii zasobów serwera matki i (w zależności od świadczonej usługi) może być realizowany na zlecenie stacji klienckiej lub serwera usług. Pojedyncza sesja, w której bierze udział stacja kliencka oraz serwer usług, inicjowana jest zawsze przez stację kliencką. Użytkownik, który chce skorzystać z danej usługi, pozyskuje informację o adresie anykastowym serwera świadczącego tę usługę. Informację taką może otrzymać, przykładowo, za pośrednictwem protokołów warstwy sesji, takich jak wykorzystywany typowo podczas sesji multikastowych protokół SAP (*Session Announcement Protocol*) [7] informowania o sesji, czy protokół SIP (*Session Initiation Protocol*) [18] zapraszania do sesji. Możliwe jest również przekazywanie informacji o usłudze w dowolny inny sposób – np. za pośrednictwem strony WWW lub poczty elektronicznej.

Uzyskawszy informację o usłudze, stacja kliencka wysyła żądanie dostępu do usługi. Żądanie to wysyła na adres anykastowy serwera usług. Zostaje ono skierowane do najbliższego (w sensie zastosowanej metryki) serwera usług.

Serwer, który odebrał żądanie dostępu od stacji klienckiej, rejestruje taką stację jako legalnego użytkownika. W razie potrzeby, przed zarejestrowaniem dokonuje autoryzacji stacji. Jeżeli usługa jest płatna, zostaje także uruchomiona odpowiednia procedura bilingowa.

Po zarejestrowaniu, serwer przesyła stacji klienckiej adres, który będzie służył do redystrybucji danych. Adres ten jest zawsze adresem unikastowym lub multikastowym, a typ adresu zależy od świadczonej usługi. Jeżeli usługa wymaga, by transmisja realizowana była indywidualnie do każdego z użytkowników, stosowana jest adresacja (a co za tym idzie, również transmisja) typu unikast. W przeciwnym wypadku stosowana jest adresacja multikastowa. Użycie adresu multikastowego (a więc i transmisji multikastowej) pozwala na efektywne rozsyłanie tego samego przekazu równocześnie do wszystkich podłączonych stacji klienckich.

Warto zauważyć, że zarówno adres anykastowy, jak i unikastowy jest adresem przeznaczonym do transmisji punkt-punkt. Użycie adresu typu unikast zamiast adresu anykastowego zapobiega jednakże ewentualnym problemom, jakie mogłyby wystąpić podczas nieoczekiwanej zmiany serwera usług, wynikającej z zadziałania procedur rutingu anykastowego (patrz [3]).

W danej podsieci może, lecz nie musi występować kilka serwerów usług. Wówczas transmisja anykastowa umożliwia realizację systemu o zwiększonej niezawodności, w którym, w przypadku awarii danego serwera usług, transmisja automatycznie jest przekierowywana do nowego serwera usług o tym samym adresie anykastowym. Wykorzystywanie, po rejestracji stacji klienta, transmisji typu unikast powoduje, iż funkcja ta byłaby niedostępna. Aby tego uniknąć, po wykryciu przez aplikację braku łączności z serwerem usług, aplikacja klienta ponawia procedurę rejestracyjną z wykorzystaniem adresacji anykastowej. Jeżeli w trakcie transmisji nastąpiła podmiana serwera usług, klient podłączy się do nowego serwera.

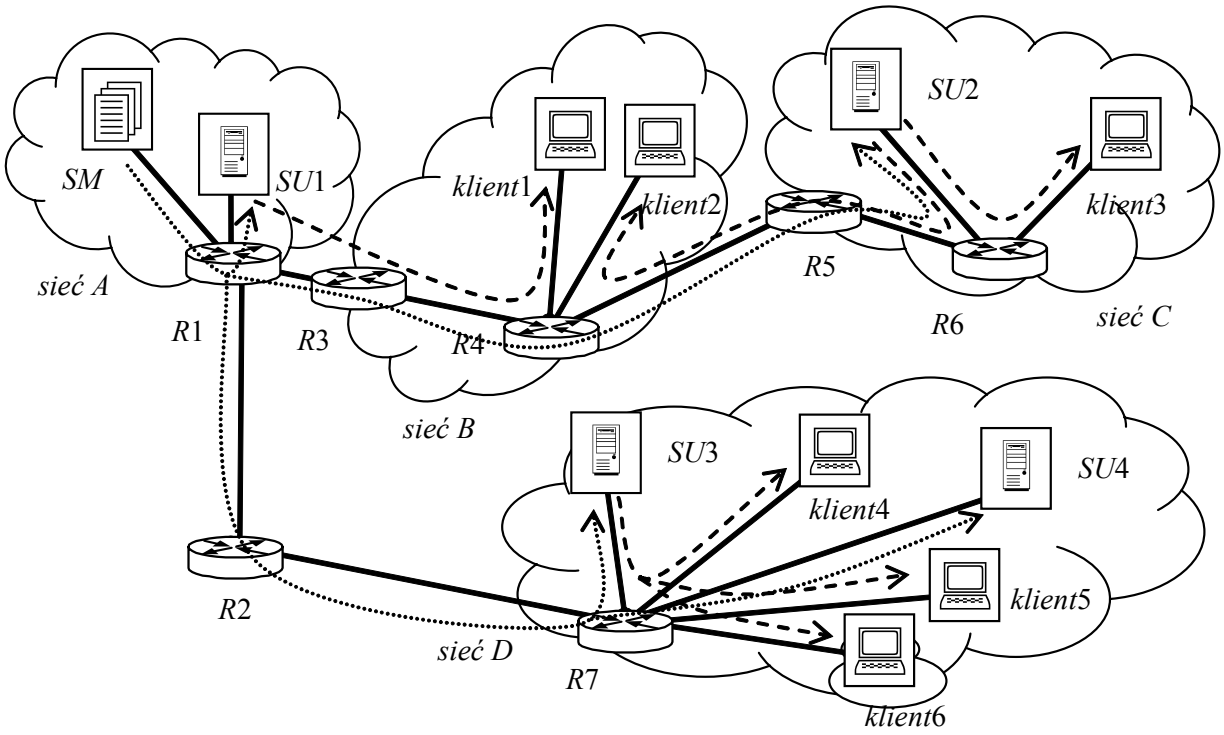
Po zakończeniu transmisji, użytkownik wyrejestrowuje się z serwera usług. Aby uniknąć sytuacji, gdy stacja kliencka, odłączając się (np. w wyniku awarii) bez wysłania komunikatu wyrejestrowania, pozostaje ciągle zarejestrowana, po upływie pewnego czasu nieaktywności (rzędu 20–30 min) rejestracja powinna samoczynnie wygasać.

Na rysunku 1 została zaprezentowana przykładowa sieć dystrybucyjna, zgodna z powyższą koncepcją. Składa się ona z jednego serwera matki *SM*, czterech serwerów usług *SU*1, …, *SU*4 oraz sześciu stacji klienckich *klient*1, …, *klient*6. Elementy te zlokalizowane są w czterech podsieciach *A*…*D*. Podsiecią (w sensie niniejszej koncepcji) może być sieć lub podsieć danego operatora lub danej instytucji, grupująca użytkowników zlokalizowanych (geograficznie) na pewnym, ograniczonym obszarze (np. jednego miasta, gminy, dzielnicy, a nawet biurowca). Może to być zatem sieć lokalna, sieć miejska lub jej fragment, sieć korporacyjna lub jej wydzielony fragment. Granica obszaru sieci jest ustalana administracyjnie przez operatora.

Serwer matka umiejscowiony został w sieci *A*. Serwer ten nie świadczy usług stacjom klienckim, a jedynie usługę multikastowej dystrybucji danych pierwotnych do serwerów usług *SU* zlokalizowanych zarówno w sieci miejscowej serwera matki (*SU*1), jak i poza nią (*SU*2, *SU*3, *SU*4). W prostych systemach dystrybucyjnych, serwer matka i serwer usług mogą zostać posadowione fizycznie na jednej maszynie (stacji roboczej), jednak w takim wypadku zawsze muszą to być dwa osobne procesy.

Stacja kliencka, która zamierza korzystać z usługi świadczonej przez system dystrybucyjny, podłącza się do najbliższego (w sensie metryki) serwera usług, wykorzystując właściwości adresacji anykastowej. Typowo, w podsieci znajduje się jeden serwer usług (sieć *C* na rys. 1). Do serwera tego podłączają się wszystkie stacje robocze zlokalizowane w danej podsieci.

Serwer usług może, ale nie musi być zlokalizowany w danej podsieci. Jeżeli w podsieci nie ma serwera usług (sieć *B* na rys. 1), to stacje klienckie zlokalizowane w tej podsieci łączą się z najbliższym (w sensie metryki) serwerem usług.



Rys. 1. Przykładowa sieć dystrybucyjna. Legenda: linia kropkowana – dystrybucja danych pierwotnych do serwerów usług, linia kreskowana – redystrybucja danych do stacji klienckich

Załóżmy, że w chwili czasu t_1 stacja kliencka *klient*1 zlokalizowana w sieci *B* podłączyła się do serwera usług *SU*1, znajdującego się w sieci *A*. Jeżeli metryka zbudowana jest na bazie topologii sieci, to w chwili czasu t_2 stacja kliencka *klient*2 z sieci *B* również podłączy się do *SU*1. Jeżeli natomiast metryka uwzględnia również bieżący stan sieci (np. obciążenie łączy), to *klient*2 równie dobrze może zostać skierowany do serwera *SU*2 z podsieci *C*. Zwróćmy uwagę, że liczba węzłów pośredniczących pomiędzy stacją kliencką a serwerem w obydwu przypadkach jest taka sama.

Z pewnych względów (np. w celu zapewnienia większej niezawodności redystrybucji danych) może być korzystne umieszczenie w danej sieci więcej niż jednego serwera usług. W sieci *D* na rysunku 1 znajdują się trzy stacje klienckie *klient*4, …, *klient*6 oraz dwa serwery usług *SU*3 i *SU*4. Typowo, jeden z serwerów usług (tu: *SU*3) jest serwerem podstawowym, do którego podłączane są stacje klienckie. Drugi natomiast (tu: *SU*4) jest serwerem rezerwowym. Dopóki serwer podstawowy *SU*3 jest sprawny, procedury rutingu anykastowego kierują wszystkie żądania klientów do tego serwera. Jeżeli serwer *SU*3 ulegnie uszkodzeniu, jego rolę przejmie serwer rezerwowy *SU*4 (jest to tzw. rezerwa nieobciążona). Dzięki usłudze adresacji anykastowej, zarówno przełączenie pomiędzy serwerem podstawowym a serwerem rezerwowym, jak i wybór serwera podstawowego dokonywane są automatycznie (w warstwie sieciowej z wykorzystaniem procedur rutingu anykastowego) i nie wymagają ingerencji operatora. Warto zauważyć, że (o ile ruting na to pozwoli) możliwy jest również wariant redundancji z rezerwą obciążoną. Wówczas stacje klienckie zlokalizowane w sieci *D* będą się podłączały do obydwu serwerów usług znajdujących się w tej sieci.

4. Rejestracja i wyrejestrowywanie serwera usług w serwerze matce

Aktualizacja zasobów w serwerach usług wymaga zarejestrowania tego serwera w serwerze matce. W omawianej koncepcji systemu dystrybucyjnego, realizowane jest to poprzez ciąg następujących operacji:

- Serwer usług wysyła do serwera matki żądanie rejestracji.
- Serwer-matka dokonuje autoryzacji serwera usług; jeśli autoryzacja zakończyła się sukcesem, realizowany jest dalszy ciąg procedury rejestracyjnej; jeżeli nie, procedura rejestracyjna ulega zakończeniu, a administrator serwera matki jest informowany o próbie nieuprawnionego podłączenia się do serwera.
- Serwer matka sprawdza, czy serwer usług jest uprawniony do świadczenia danej usługi lub pakietu usług administrowanych przez serwer

matkę. W razie niepowodzenia, administrator serwera matki jest informowany o próbie nieuprawnionego dostępu do usługi.

- Serwer matka rejestruje nowy serwer usług w ruterach anykastowych, które aktualizują swoje tablice.
- Serwer usług otrzymuje podstawowe parametry konfiguracyjne sesji, w tym adres anykastowy świadczonej usługi i adres multikastowy, pod którym będzie realizowana dystrybucja danych pierwotnych. Jeżeli będzie wymagana autoryzacja serwera usług posługującego się danym adresem anykastowym w ruterach, to (w zależności od metody autoryzacji) serwer matka dostarcza również niezbędne klucze, hasła bądź sygnatury.
- Jeżeli aktualizacja realizowana jest z szyfrowaniem, to serwer matka udostępnia również bieżący klucz deszyfrujący.

Rejestracja w serwerze matce wymaga cyklicznego odświeżania, gdyż po pewnym czasie wygasa. Czas wygaśnięcia rejestracji T_w jest jednym z parametrów konfiguracyjnych sesji, otrzymywanych przez serwer usług na etapie rejestracji. Odświeżanie powinno być realizowane nie wcześniej niż po upłynięciu połowy tego czasu. Serwer usług odnawiający rejestrację losuje czas wysłania kolejnego żądania rejestracji z przedziału obustronnie domkniętego $<0{,}5T_w, 0{,}9T_w>$. Odświeżanie rejestracji może się odbywać w trybie pełnym (wymagane jest wówczas przejście pełnego cyklu rejestracji) lub uproszczonym. W trybie uproszczonym nie ma potrzeby np. ponownej rejestracji serwera usług w ruterach anykastowych.

Takie odświeżanie rejestracji pozwala łatwo wykryć awarię systemu dystrybucji. Pozwala również w sposób naturalny wyrejestrować serwery usług, które odłączyły się bez wyrejestrowania.

Po zakończeniu rejestracji, możliwy jest odbiór przekazów od serwera matki. Stan taki trwa dopóty, dopóki serwer usług się nie wyrejestruje (lub nie zostanie wyrejestrowany ze względu na wygaśnięcie rejestracji) lub serwer matka nie zakończy pracy systemu.

Wyrejestrowanie serwera usług z serwera matki odbywa się z wykorzystaniem specjalnego komunikatu wyrejestrowania. Komunikat ten powinien być uwierzytelniony. Wyrejestrowywanie serwera usług wymaga między innymi wprowadzenia zmian na liście aktualnych serwerów usług oraz zakończenia bilingu dla tego serwera.

Wyrejestrowywanie serwera usług wymaga również wykonania pewnych czynności koordynowanych przez serwer matkę, a realizowanych poza tym serwerem. Czynnością taką jest np. aktualizacja tablic rutingu anykastowego. Stacja przestaje być zarejestrowanym serwerem danej usługi i żądania dostępu do usługi, wysyłane przez stację kliencką, powinny być kierowane do innego serwera.

Po każdym wyrejestrowaniu serwera usług powinna nastąpić zmiana klucza szyfrującego. Zabezpiecza to przed nieuprawnionym odbiorem danych pierwotnych (z pominięciem np. procedur bilingowych) przez wyrejestrowany serwer usług. Należy bowiem pamiętać, że członkostwo w grupie multikastowej jest anonimowe i każda stacja, która podłącza się do grupy, może odbierać dane rozsyłane na adres grupy. Jeżeli bezpieczeństwo przekazu tego wymaga, zmiana klucza szyfrującego może następować również w trakcie transmisji.

5. Dystrybucja danych pierwotnych

Do dystrybucji danych pierwotnych pomiędzy serwerem matką a serwerami usług wykorzystywany jest typowo niezawodny multikastowy protokół transportowy. Ponieważ wymagania poszczególnych usług mogą być różne, to zbudowanie jednego uniwersalnego protokołu transportowego spełniającego wymagania różnorodnych usług i realizujący operacje protokołowe efektywnie jest praktycznie niemożliwe [6]. Dlatego też projektując system dystrybucji należy dobrać odpowiedni protokół transportowy do danej usługi. W ramach prowadzonych prac dokonano analizy szeregu istniejących niezawodnych multikastowych protokołów transportowych pod kątem zastosowania ich w systemie dystrybucji.

W wyniku analizy wskazano cztery protokoły, które są w stanie spełnić wymagania wielu usług. Są to:

1) protokół PGM,
2) protokół NORM,
3) protokół ALC,
4) protokół T-RMP.

Protokół PGM [5, 19] jest uniwersalnym multikastowym protokołem transportowym, przeznaczonym do niezawodnej transmisji danych. Protokół ten jest obecnie wdrażany przez znaczących producentów systemów operacyjnych i sprzętu sieciowego.

W celu zapewnienia niezawodności przekazu, protokół PGM wykorzystuje retransmisję selektywną uszkodzonych lub zagubionych pakietów danych. Są one wysłane w reakcji na żądania odbiornika wysyłane w potwierdzeniach negatywnych NAK (*negative acknowledgment*). PGM dopuszcza również użycie metody FEC (*Forward Error Correction*).

Zastosowanie potwierdzeń negatywnych eliminuje implozję potwierdzeń w przypadku transmisji bezbłędnej, jednakże może ona wystąpić w przypadku błędu transmisji. Aby tego uniknąć, protokół PGM dokonuje agregacji i tłumienia potwierdzeń NAK.

Agregacja potwierdzeń wykorzystuje rutery multikastowe wspierające protokół PGM, tzw. rutery (lub elementy sieciowe) PGM NE (*PGM-capable Network Elements*). Protokół PGM nie wymaga przy tym, aby wszystkie rutery zlokalizowane wzdłuż drzewa dystrybucji multikastowej wspierały protokół PGM. Ruter PGM NE odbiera potwierdzenia NAK od odbiorników znajdujących się w dół drzewa dystrybucji, generuje jedno wspólne potwierdzenie NAK i przesyła je do systemu nadrzędnego.

Tłumienie potwierdzeń negatywnych opiera się na spostrzeżeniu, iż w transmisji multikastowej wystarczy, by o błędzie poinformował jeden (i tylko jeden) z odbiorników wykrywających zagubiony lub uszkodzony pakiet. W razie wystąpienia błędu, odbiornik otrzymujący NAK pochodzący od innej stacji rezygnuje z wysłania potwierdzenia negatywnego w reakcji na ten sam błąd.

Protokół PGM nie realizuje zapobiegania przeciążeniom, w tym celu zwykle wykorzystywany jest, współpracujący z PGM, blok funkcjonalny PGMCC (nazwa własna; nieoficjalne rozwinięcie skrótu to „PGM *(Pragmatic General Multicast)* Congestion Control"). PGMCC należy do klasy tzw. protokołów sprzyjających TCP i, jako taki, emuluje mechanizm zapobiegania przeciążeniom stosowany w protokole TCP. Okno przeciążeniowe, charakterystyczne dla TCP, jest tu emulowane za pomocą mechanizmu *token bucket*, który ogranicza strumień danych wysyłanych przez protokół PGM.

Protokół NORM (*Negative-acknowledgment (NACK)-Oriented Reliable Multicast*) [1, 2] został zaprojektowany do niezawodnej dystrybucji danych masowych od jednego nadajnika do wielu odbiorców równocześnie. Przewidywane jest w przyszłości rozszerzenie specyfikacji protokołu o możliwość rozsyłania danych przez wiele nadajników w ramach pojedynczej sesji.

Protokół NORM należy do grupy tzw. protokołów o modułowej architekturze, opracowanej przez grupę roboczą IETF *rmt* (*Reliable Multicast Transport*). Protokoły te budowane są z modułów (*building block*), wspólnych dla szeregu rozwiązań. Poszczególne moduły muszą spełniać wymagania określone przez RFC 3048 [23].

Protokół NORM wykorzystuje dwa bloki funkcjonalne (moduły) opracowane w ramach grupy *rmt*:

1) blok NORM [2],
2) blok FEC [10, 11].

Do zapobiegania przeciążeniom stosowany jest mechanizm NORM-CC, zbudowany na bazie koncepcji bloku TFMCC (*TCP-Friendly Multicast Congestion Control*) [24].

Zapewnienie niezawodności w protokole NORM realizowane jest z wykorzystaniem retransmisji uszkodzonych bądź zagubionych pakietów. Pakiety te wysyłane są przez nadajnik na żądanie odbiornika. Odbiornik wysyła żądania retransmisji pakietów w pakietach potwierdzeń negatywnych NACK (*negative-acknowledgment*). Dodatkowo, aby zwiększyć efektywność protokołu, stosowany jest algorytm FEC. Domyślnie algorytm FEC wykorzystywany jest w przypadku wysyłania pakietów danych podlegających retransmisji (tj. wysyłanych w reakcji na pakiety NACK). W przypadku dużych grup multikastowych, algorytm FEC stosowany jest także w podstawowych pakietach danych, co znacząco zwiększa skalowalność protokołu (zmniejsza się liczba potwierdzeń negatywnych NACK).

Aby uniknąć implozji potwierdzeń negatywnych, w protokole NORM zastosowano tłumienie potwierdzeń NACK wysyłanych w reakcji na ten sam błąd. Specyfikacja protokołu NORM przewiduje również wsparcie ze strony ruterów, które to realizowałyby agregację potwierdzeń NACK.

Oprócz potwierdzeń negatywnych protokół NORM wykorzystuje także potwierdzenia pozytywne ACK. Są one generowane w odpowiedzi na stosowne polecenia nadajnika i wykorzystywane głównie przez mechanizm zapobiegania przeciążeniom. Potwierdzenia pozytywne podlegają takim samym procedurom agregacji, co potwierdzenia negatywne.

W protokole NORM dane wysyłane przez aplikację podzielono (ze względu na rodzaj zawartości) na trzy kategorie:

1) dane będące określonym fragmentem pamięci operacyjnej nadajnika (*NORM_OBJECT_DATA*),
2) dane, które pochodzą z określonego pliku składowanego w systemie plików nadajnika (*NORM_OBJECT_FILE*),
3) dane tworzące ciągły, nieskończony strumień (*NORM_OBJECT_STREAM*).

Rozróżnienie rodzaju przesyłanej zawartości, dokonywane w protokole NORM, pozwala odbiornikowi na wybranie stosownych procedur obsługi i przygotowanie się do odbioru specyficznych danych. Wiąże się to np. z zarezerwowaniem odpowiednich buforów odbiorczych i przygotowaniem urządzeń wejścia/wyjścia. Przykładowo, wskazanie, iż dane przenoszą zawartość pliku, pozwala na poinformowanie aplikacji odbiorczej, że ilość przesyłanych danych będzie stosunkowo duża (zazwyczaj większa, niż w przypadku transmisji fragmentów pamięci operacyjnej).

Wykorzystanie trzeciego z typów danych (dane strumieniowe) wskazuje z kolei, iż przesyłane dane nie mają z góry określonego limitu lub jest on nieznany w chwili rozpoczęcia wysyłania danych (np. transmitowany jest strumień będący wynikiem przetwarzania aplikacji realizującej w czasie rzeczywistym wyszukania danych w bazie danych). W przypadku tego typu transmisji możliwe jest opóźnione przyłączanie się odbiorników (tj. istnienie odbiorników przyłączających się już w trakcie trwania transmisji).

W przypadku proponowanego systemu dystrybucji danych pierwotnych, dla wielu typów danych najbardziej odpowiednie będzie wykorzystanie trybu transmisji

właściwego dla przesyłania plików (*NORM_OBJECT_FILE*). W przypadku danych pierwotnych pochodzących ze źródeł strumieniowych, wskazane jest zastosowanie protokołu NORM w trybie strumieniowym. Należy jednak uwzględnić fakt, iż protokół NORM nie gwarantuje w tym przypadku spełnienia warunków transmisji w czasie rzeczywistym, koniecznej w przypadku wielu mediów strumieniowych.

Protokół ALC (*Asynchronous Layered Coding*) [8] został zaprojektowany jako w pełni skalowalny multikastowy protokół do niezawodnej transmisji danych masowych (o objętości od kilku kilobajtów do kilkuset gigabajtów). Ma on, zgodnie z założeniami projektowymi, umożliwiać transmisję do milionów odbiorców.

Protokół ALC wykorzystuje dwa bloki funkcjonalne:

1) LCT (Layered Coding Transport) [9],
2) FEC [10, 11].

Implementuje on również zgodny z RFC 2357 [14] blok przeciwdziałania przeciążeniom [8].

Typowo w protokole ALC dane przesyłane są w postaci obiektów. Obiektem może być np. strona WWW, plik danych. W przypadku przesyłania szeregu obiektów odbiornik ma możliwość wyboru obiektów, które go interesują. Oprócz transmisji obiektów specyfikacja protokołu przewiduje transmisję strumienia danych.

Transmisja każdego obiektu (lub strumienia) może być realizowana z wykorzystaniem jednej lub kilku warstw. W przypadku zastosowania wielu warstw każda warstwa charakteryzuje się inną przepływnością binarną. Odbiór danego obiektu jest możliwy zarówno poprzez odbiór tylko jednej podstawowej warstwy, jak i kilku warstw. Im większa ilość warstw jest poprawnie odbierana przez odbiornik, tym szybciej odbiornik jest w stanie odebrać dany obiekt [8]. Ilość warstw, jakie odbiera dany odbiornik, zależy od mechanizmów przeciwdziałania przeciążeniom. Mechanizmy te powinny umożliwić zarówno dużą szybkość odbioru obiektów, jak i zapewnić dobre współistnienie protokołu ALC z innymi protokołami w sieci.

Protokół ALC realizuje niezawodną transmisję danych dzięki zastosowaniu algorytmu FEC, co powoduje wyeliminowanie pakietów potwierdzeń zarówno pozytywnych, jak i negatywnych. Z drugiej strony, stosowany w protokole ALC mechanizm przeciwdziałania przeciążeniom pracuje bez sprzężenia zwrotnego – a zatem nie wymaga żadnych pakietów potwierdzających. Brak pakietów potwierdzeń w mechanizmie przeciwdziałania przeciążeniom i mechanizmach zapewniających niezawodność pozwala na zbudowanie w pełni skalowalnego systemu dystrybucji multikastowej.

Protokół ALC ze względu na brak potwierdzeń nie wymaga kanału zwrotnego. Może on zatem pracować również w sieciach niesymetrycznych, np. satelitarnych bez kanału zwrotnego.

Na bazie protokołu ALC zbudowano protokół FLUTE (*File Delivery over Unidirectional Transport*) [16] do dystrybucji plików w sieci IP. Protokół ten zakłada jednokierunkową transmisję od jednego nadawcy do wielu odbiorców. Ponieważ w warstwie transportowej wykorzystywany jest protokół ALC, FLUTE nie wymaga kanału zwrotnego od odbiorców.

Specyfikacja protokołu FLUTE określa mechanizmy sygnalizacyjne, które przekazują dane niezbędne do prawidłowego odtworzenia pliku w systemie końcowym. Są to m.in. URI (*Uniform Resource Identifier*) danego pliku, typ pliku w formacie MIME (*Multipurpose Internet Mail Extensions*), rozmiar pliku, metodę kompresji i kodowania pliku, dane niezbędne do uwierzytelnienia i deszyfrowania pliku. Zastosowanie protokołu FLUTE w proponowanym systemie aktualizacji zasobów serwerów może znacząco uprościć budowę systemu dystrybucyjnego wykorzystującego protokół ALC.

Pomimo iż protokoły NORM i ALC są wskazywane jako możliwe do wykorzystania do transmisji danych strumieniowych, brak w nich mechanizmów wspomagających transmisję czasu rzeczywistego oraz synchronizację otrzymywania danych przez wszystkich odbiorców. Jeżeli zatem aktualizacja zawartości serwerów usług z pewnych względów musi być realizowana z tym samym czasie i z zapewnieniem synchronizacji odbioru (np. gdy serwery usług dokonują redystrybucji danych giełdowych), należy zastosować specjalizowany niezawodny protokół transportowy. Przykładem takiego rozwiązania może być eksperymentalny protokół T-RMP (*Timed Reliable Multicast Protocol*).

Protokół T-RMP [4, 15] jest rozszerzoną wersją protokołu RMP (*Reliable Multicast Protocol*) [4, 22], opracowanego pod kątem niezawodnej, multikastowej transmisji danych od wielu źródeł do wielu odbiorców (*M*-do-*N*). W stosie protokołowym zajmuje on górną podwarstwę warstwy transportowej, podczas gdy dolną podwarstwę zajmuje protokół UDP (*User Datagram Protocol*). Protokół RMP dostarcza kilku poziomów jakości transmisji danych (*QoS levels*), począwszy od poziomu bez zapewnienia gwarantowanej jakości usług (jak np. w protokole UDP), a kończąc na poziomie niezawodnego dostarczania danych w kolejności z wszystkich nadających źródeł [4].

RMP realizuje retransmisję selektywną, a sygnalizacja błędów odbywa się poprzez selektywne potwierdzenia ACK. Aby zapobiec, niemal nieuchronnej w takiej sytuacji, implozji komunikatów potwierdzeń, zastosowano tutaj ideę pierścienia z przekazywaniem uprawnień (*token ring*). Odbiorniki multikastowe tworzą pierścień wirtualny, a ten spośród nich, który w danym momencie posiada uprawnienie (*token*), potwierdza w imieniu grupy prawidłowy odbiór danych. W ten sposób, sys-

tem zapewnia niezawodną dystrybucję danych z maksymalnym opóźnieniem jednego pełnego obiegu uprawnienia, co sprawia, że protokół RMP jest stosunkowo słabo skalowalny.

W razie awarii pierścienia, odbiorniki przerywają pracę i synchronizują się z najlepszym (tj. tym, który odebrał najwięcej danych), wysyłając indywidualne żądania retransmisji brakujących pakietów (negatywne potwierdzenia NACK). Dalsza transmisja jest realizowana dopiero po zsynchronizowaniu się wszystkich odbiorników i odtworzeniu pierścienia.

Protokół T-RMP dodaje do funkcjonalności protokołu RMP gwarancje terminowości dostarczanych danych (z dokładnością do opóźnień transmisyjnych wynikających z czasu propagacji i buforowania [4]). Osiągnięto to poprzez synchroniczne przekazywanie uprawnienia, pobudzane zegarem (zamiast poprzedniego, asynchronicznego, pobudzanego pakietami danych). Wymaga to jednak globalnej synchronizacji zegarów we wszystkich odbiornikach. Może być ona realizowana np. za pomocą jednego z protokołów synchronizacji czasu lub poprzez GPS [4]. W T-RMP poprawiono również skalowalność systemu dystrybucji w stosunku do RMP, wprowadzając hierarchiczne zarządzanie transmisją danych.

6. Podsumowanie

W artykule zaproponowano koncepcję systemu aktualizacji zasobów serwerów korzystających z adresacji anykastowej. Dzięki zastosowaniu hierarchicznej struktury zarządzania systemem (działającej na trzech poziomach – serwera matki, serwerów usług, stacji klienckich) może być on stosowany do dystrybucji danych masowych na dużym obszarze geograficznym. Do transmisji pomiędzy pierwszym a drugim poziomem hierarchii wykorzystano niezawodną transmisję multikastową, co daje możliwość efektywnej aktualizacji danych we wszystkich serwerach usług równocześnie.

Proponowana koncepcja systemu aktualizacji zasobów serwerów korzystających z adresacji anykastowej nie jest rozwiązaniem zamkniętym. Może być łatwo dostosowana do spełniania innego typu zadań, niż tylko dystrybucja danych masowych na dużym obszarze geograficznym.

Proponowana metoda aktualizacji zasobów serwerów korzystających z adresacji anykastowej może być, w całości lub w wersji uproszczonej, zastosowana do komunikacji w systemie obliczeń rozproszonych. Do komunikacji pomiędzy serwerami usług (pełniącymi wówczas funkcje wykonawcze w systemie obliczeniowym) należałoby wówczas zastosować protokół ALC, realizujący niezawodne przesyłanie danych traktowanych jako poszczególne obiekty. Można również stosować protokół NORM w trybie pozwalającym na przesyłanie danych będących określonym fragmentem pamięci operacyjnej nadajnika.

Innym przykładem może być zastosowanie proponowanej koncepcji do dystrybucji informacji multimedialnej w czasie rzeczywistym – np. w usłudze telewizji internetowej. Serwer matka pełniłby tutaj rolę dostawcy treści przekazu multimedialnego (*content provider*). Serwer usług natomiast pełniłby rolę specjalizowanego serwera proxy dla usługi czasu rzeczywistego. W bardziej rozbudowanych systemach mógłby również wykonywać zadania dostawcy usługi multimedialnej (*service provider*). Multikastowa dystrybucja danych pierwotnych (tu: audycji telewizyjnej) z serwera matki do serwera usług byłaby wówczas realizowana z wykorzystaniem protokołu RTP, zaprojektowanego pod kątem transmisji czasu rzeczywistego.

Literatura

[1] Adamson B., Bormann C., Handley M., Macker J.: *Negative-acknowledgment (NACK)-Oriented Reliable Multicast (NORM) Protocol*. RFC 3940, IETF, November 2004

[2] Adamson B., Bormann C., Handley M., Macker J.: *Negative-Acknowledgment (NACK)-Oriented Reliable Multicast (NORM) Building Blocks*. RFC 3941, IETF, November 2004

[3] Chodorek A., Chodorek R.R., Pach A.R.: *Wdrażanie usług korzystających z adresacji anykast – dyskusja problemów i* zagrożeń. Telekomunikacja Cyfrowa – Technologie i usługi, tom 7, 2005

[4] Chodorek R.R., Pach A.R.: *Transmisja multikastowa w sieciach IP*. Kraków, Wydawnictwo FPT 2003

[5] Chodorek R.R.: *Analiza multikastowej transmisji danych z wykorzystaniem protokołu PGM*. Rozdział w książce: praca zbiorowa pod red. A. Kwietnia i A. Grzywaka Wysokowydajne sieci komputerowe. Zastosowanie i bezpieczeństwo, Warszawa, WKŁ 2005

[6] Handley M., Floyd S., Whetten B., Kermode R., Vicisano L., Luby M.: *The Reliable Multicast Design Space for Bulk Data Transfer*. RFC 2887, IETF, sierpień 2000

[7] Handley M., Perkins C., Whelan E.: *Session Announcement Protocol*. RFC 2974, IETF, October 2000

[8] Luby M., Gemmell J., Vicisano L., Rizzo L., Crowcroft J.: *Asynchronous Layered Coding (ALC) Protocol Instantiation*. RFC 3450, IETF, December 2002

[9] Luby M., Gemmell J., Vicisano L., Rizzo L., Handley M., Crowcroft J.: *Layered Coding Transport (LCT) Building Block*. RFC 3451, IETF, December 2002

[10] Luby M., Vicisano L., Gemmell J., Rizzo L., Handley M., Crowcroft J.: *Forward Error Correction (FEC) Building Block*. RFC 3452, IETF, December 2002

[11] Luby M., Vicisano L., Gemmell J., Rizzo L., Handley M., Crowcroft J.: *The Use of Forward Error Correction (FEC) in Reliable Multicast*. RFC 3453, IETF, December 2002

[12] Luciani J., Armitage G., Halpern J., Doraswamy N.: *Server Cache Synchronization Protocol (SCSP)*. RFC 2334, IETF, April 1998

[13] Luotonen A., Altis K.: *World-Wide Web Proxie*. Computer Networks and ISDN Systems, Vol. 27, No. 2, 1994

[14] Mankin A., Romanow A., Bradner S., Paxson V.: *IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols*. RFC 2357. IETF, June 1998

[15] Maxemchuk N.F.: *Reliable Multicast with Delay Guarantees*. IEEE Communications Magazine, Vol. 40, No. 9, September 2002

[16] Paila T., Luby M., Lehtonen R., Roca V., Walsh R.: *FLUTE - File Delivery over Unidirectional Transport*. RFC 3926, IETF, October 2004

[17] Partridge C., Mendez T., Milliken W.: *Host Anycasting Service*. RFC 1546, IETF, November 1993

[18] Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J., Sparks R., Handley M., Schooler E.: *SIP: Session Initiation Protocol*. RFC 3261, IETF, June 2002

[19] Speakman T., Crowcroft J., Gemmell J., Farinacci D., Lin S., Leshchiner D., Luby M., Montgomery T., Rizzo L., Tweedly A., Bhaskar N., Edmonstone R., Sumanasekera R., Vicisano L.: *PGM Reliable Transport Protocol Specification*. RFC 3208, IETF, December 2001

[20] Wessels D., Claffy K.: *Application of Internet Cache Protocol (ICP), version 2*. RFC 2187, IETF, September 1997

[21] Wessels D., Claffy K.: *Internet Cache Protocol (ICP), version 2*. RFC 2186, IETF, September 1997

[22] Whetten B., Montgomery T., Kaplan S.: *A High Performance, Totally Ordered Multicast Protocol*. Theory and Practice in Distributed Systems, Springer Verlag LNCS 938, 1994

[23] Whetten B., Vicisano L., Kermode R., Handley M., Floyd S., Luby M.: *Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer*. RFC 3048, IETF, January 2001

[24] Widmer J., Handley M.: *TCP-Friendly Multicast Congestion Control (TFMCC):Protocol Specification*. INTERNET-DRAFT, Work in progress, IETF, July 2003. <draft-ietf-rmt-bb-tfmcc-02.txt>

Praca naukowa finansowana ze środków Komitetu Badań Naukowych w latach 2003–2005 jako projekt badawczy nr 4 T11D 015 24.

Robert Chodorek ukończył studia na Wydziale Elektrotechniki, Automatyki i Informatyki Politechniki Świętokrzyskiej w Kielcach w roku 1990. Stopień doktora nauk technicznych uzyskał na Wydziale Elektrotechniki, Automatyki i Elektroniki Akademii Górniczo-Hutniczej w Krakowie w roku 1996. Obecnie jest pracownikiem Katedry Telekomunikacji AGH. Zainteresowania zawodowe obejmują: sieci teleinformatyczne, protokoły komunikacyjne, badanie wydajności protokołów, specyfikacja protokołów, aplikacje multimedialne.

Agnieszka Chodorek ukończyła studia na Wydziale Elektrotechniki, Automatyki i Informatyki Politechniki Świętokrzyskiej w Kielcach w roku 1991. Uzyskała stopień naukowy doktora inżyniera w 2001 roku. Obecnie jest pracownikiem Samodzielnego Zakładu Telekomunikacji i Fotoniki Politechniki Świętokrzyskiej. Zainteresowania zawodowe obejmują: analizę ruchu w sieciach telekomunikacyjnych, sieci teleinformatyczne, multimedia.

Andrzej Ryszard Pach ukończył Wydział Elektrotechniki, Automatyki i Elektroniki AGH w roku 1975, w r. 1977 doktoryzował się na AGH, a w roku 1990 uzyskał stopień doktora habilitowanego na Wydziale Elektroniki Politechniki Warszawskiej. Zatrudniony jest obecnie na stanowisku profesora zwyczajnego w Katedrze Telekomunikacji AGH, w której pełni funkcję kierownika. Wcześniej był prodziekanem Wydziału EAiE.

Główne zainteresowania naukowe związane są z sieciami telekomunikacyjnymi oraz systemami informacyjnymi. Autor ponad stu publikacji naukowych z zakresu protokołów komunikacyjnych, modelowania i analizy sieci komputerowych, sieci szerokopasmowych z integracją usług. Aktywnie uczestniczy w projektach europejskich IST, ACTS, COST i COPERNICUS. Członek komitetów programowych konferencji międzynarodowych. Konsultant firm państwowych i prywatnych w zakresie nowoczesnej telekomunikacji.

Współzałożyciel i wiceprezydent Fundacji Postępu Telekomunikacji, przewodniczący IEEE Communications Society Chapter.