STUDIA INFORMATICA 2000
Volume 21 Number 1 (39)

Cezary ŻÓŁKOWSKI, Jacek WYTRĘBOWICZ
Politechnika Warszawska, Instytut Informatyki

ALGORYTMY REZERWACJI PASMA W SIECIACH LOKALNYCH

Streszczenie. Praca zawiera krótki opis protokołów rezerwacji zasobów RSVP i ST2 stosowanych w WAN, oraz dyskusję pięciu protokołów zaprojektowanych do stosowania w popularnie używanych sieciach LAN.

BANDWIDTH RESERVATION ALGORITHMS FOR LOCAL AREA NETWORKS

Summary. The paper contains a short description of two resource reservation internet protocols i.e., RSVP and ST2. Its main part describes five protocols dedicated to widely used LANs.

1. Wstęp

Od wielu lat obserwujemy gwałtowny rozwój sieci komputerowych w sensie nie tylko liczby użytkowników, ale i różnorodności oferowanych usług. Coraz więcej usług wymaga rezerwacji pasma przenoszenia w celu zagwarantowania jakości oczekiwanej przez użytkowników. W niektórych zastosowaniach zagwarantowanie pasma jest niezbędnym warunkiem istnienia danej usługi, przykładem może tu być przesyłanie głosu, czy sterowanie w czasie rzeczywistym.

Wybierając właściwego operatora sieci rozległej czy miejskiej, możemy zapewnić sobie dostęp do protokołów rezerwacji pasma oferowanych przez danego operatora. Budując sieć lokalną, możemy wybrać właściwą technologię, np. ATM, która dostarcza takich możliwości. Lecz co zrobić, gdy już dysponujemy siecią lokalną, którą chcemy wzbogacić o nowe możliwości?

Olbrzymia większość ośrodków posiada sieci lokalne w technologii Ethernet lub Token Ring. Wymiana ich na sieć ATM może okazać się zbyt kosztowna. Warto więc rozważyć możliwości zainstalowania mechanizmu rezerwacji pasma, który pracowałby w istniejącym środowisku i kontrolował ruch pakietów generowanych przez powszechnie dziś używane protokoły TCP i UDP. Pomyślne rezultaty takiego rozwiązania pozwoliłyby na przeniesienie tego mechanizmu na niższe warstwy w stosie używanych protokołów, np. jako nakładka na protokół IP.

Celem niniejszego artykułu nie jest prezentacja gotowego rozwiązania, lecz dyskusja możliwych algorytmów, które można by zastosować przy budowie takiego mechanizmu. Rozpatrywane algorytmy sformułowaliśmy jako protokoły:

1. Protokół z zarządcą. Zarządca to wyróżniony komputer decydujący, które rezerwacje dojdą do skutku. Jest to rozwiązanie dla idealnej sieci.
2. Protokół z zarządcą o podwyższonej niezawodności. Jest on odporny na ginięcie pakietów oraz awarie komputerów z wyjątkiem zarządcy.
3. Protokół zdecentralizowany bez zarządcy dla idealnej sieci.
4. Protokół zdecentralizowany o podwyższonej niezawodności. Jest on zbudowany na podstawie rozproszonego algorytmu rezerwacji zasobów Ricarta Agrawali (1981) - dla sieci gubiącej pakiety oraz komputerów ulegających awariom.
5. Protokół o strukturze pierścienia logicznego.

W ramach naszych wcześniejszych prac protokoły te wyspecyfikowaliśmy w języku Estelle w celu ich analizy. Analiza ta prowadzona była poprzez symulację z wykorzystaniem pakietu EDT (Estelle Development Toolset) [1]. Ponadto protokoły 1, 2 i 4 zaimplementowaliśmy w języku C. Specyfikacja tych protokołów w Estelle i ich źródła w C oraz wyniki testów są umieszczone w [2].

Niniejszy artykuł jest pewnego rodzaju podsumowaniem tych prac. Omówimy tu istotę tych protokołów ze szczególnym uwzględnieniem protokołów 2 i 4. Omówienie to poprzedzone jest krótką prezentacją protokołów rezerwacji pasma dedykowanych dla zastosowań w sieciach globalnych, tj. RSVP i ST2.

2. Protokoły rezerwacji w sieciach WAN

Niniejszy rozdział ma na celu zobrazowanie rozwiązań problemu rezerwacji pasma w otoczeniu sieci lokalnych. Na ile rozwiązania te są adaptowalne w sieci lokalnej? Czy możliwe jest sprzęgnięcie protokołów rezerwacji pracujących w obu tych sieciach?

2.1. Resource ReSerVation Protocol (RSVP)

Resource ReSerVation Protocol [3] [5] jest protokołem sygnalizacyjnym, którego zadaniem jest dostarczać parametry wymaganej transmisji routerom pośredniczącym w sieci WAN. Określono dwie klasy parametrów transmisji: usługa gwarantowanej jakości, oraz usługa o kontrolowanym obciążeniu.

Usługa gwarantowanej jakości zapewnia:

- zadeklarowane pasmo przenoszenia,
- opóźnienie na całej drodze pakietu w określonych granicach oraz
- brak gubienia datagramów należących do utworzonego strumienia.

Zakłada się, że routery biorące udział w transmisji nie ulegają awarii i nie zmieniają trasy w czasie trwania połączenia. Usługa gwarantowana oparta jest na modelu przepływu, który określa opóźnienie paczki pakietów w funkcji parametrów strumienia, takich jak: r - szybkość transmisji tego strumienia oraz b - długość w bajtach paczki, przy założeniu rzeczywistej szybkości łącza R. Opóźnienie to wynosi b/R.

Aby aplikacja uzyskała usługę gwarantowaną, routery muszą sprawdzać generowany ruch, czy odpowiada on zadeklarowanym parametrom strumienia (r,b). Obsługiwane są dwa typy dozoru:

1. Sprawdzanie na końcach ścieżki (nadawca-odbiorca). Dozór ten ma na celu sprawdzenie, czy przychodzący ruch do stacji końcowej spełnia założenia o parametrach (r,b) danego strumienia.
2. Kształtowanie strumienia w węzłach pośrednich. Kształtowanie oznacza przywrócenie strumieniowi założonych parametrów we wnętrzu sieci.

Nie zawsze możliwe jest kształtowanie strumienia (np. z powodu braku wolnych buforów). W takim przypadku nadmiarowe datagramy podlegają usłudze najlepszego możliwego dostarczenia. Reasumując:

- Każda aplikacja definiuje strumień przy użyciu Tspec, tj. trzech parametrów: p, b, i r.
- Dozór musi być wykonany na końcach ścieżki, aby zapewnić, że aplikacja stosuje się do zadeklarowanego Tspec.
- Kształtowanie strumienia powinno być wykonane w węzłach pośrednich routerów, aby zapewnić, że ruch na całej ścieżce zgodny jest z Tspec.
- Jeżeli ruch przekracza Tspec, to datagramy podlegają usłudze najlepszego możliwego dostarczenia.
- Jeśli ruch zgodny jest z Tspec, to usługa gwarantowana zapewnia, że żadne datagramy nie są gubione, i że opóźnienie początek-koniec jest ograniczone przez przybliżony wzór $b/R+Ctot/R+Dtot$, gdzie b - długość paczki bajtów, R - zapewniona szybkość łącza, Ctot -

suma opóźnień w routerach zależnych od szybkości ruchu, Dtot - suma opóźnień w routerach niezależnych od szybkości ruchu.

Usługa kontrolowanego obciążenia jest bardzo podobna do usługi najlepszego możliwego dostarczenia w przypadku niewielkiego obciążenia łączy. Różnica polega na tym, że aplikacja musi poinformować routery o parametry strumienia (b, r), gdzie b-długość paczki w bajtach, r-szybkość w bajtach/sekundę, a czas rozliczenia paczki bajtów strumienia wynosi b/r. Jeżeli routery funkcjonują właściwie, to aplikacja odbierająca strumień o kontrolowanym obciążeniu może założyć, że:

1. Większość pakietów będzie dostarczona przez sieć do stacji końcowych. Procent niedostarczonych pakietów musi ściśle odzwierciedlać stopę błędów transmisji.
2. Opóźnienie całkowite większości pakietów będzie zbliżone do minimalnego opóźnienia (opóźnienie propagacji sygnału + stały czas przetwarzania w routerach) uzyskanego z kilku ostatnich dostarczonych pakietów.

Usługa kontrolowanego obciążenia bazuje na modelu przepływu, z którego wynika, że jeśli każdy router ma do dyspozycji pasmo łącza większe niż suma prędkości r strumieni, to jest możliwe obsłużenie wielu takich strumieni bez pogarszania jakości każdego z nich. Routery mogą stosować w tym celu multipleksowanie statystyczne

RSVP jest zorientowany na odbiorcę, tzn. zamówienie na zasoby jest generowane przez stację odbierającą i przekazywane demonowi RSVP na stacji. Demon RSVP w odbiorniku przekazuje zamówienie demonowi RSVP na routerze, który jest na ścieżce zwrotnej do nadawcy. W ten sposób zamówienie przenosi się na kolejne routery i ustawiają się parametry modułów klasyfikacji i szeregowania pakietów we wszystkich węzłach pośrednich, co jest celem tego protokołu.

2.2. Protokół Internet Stream Protocol - ST2

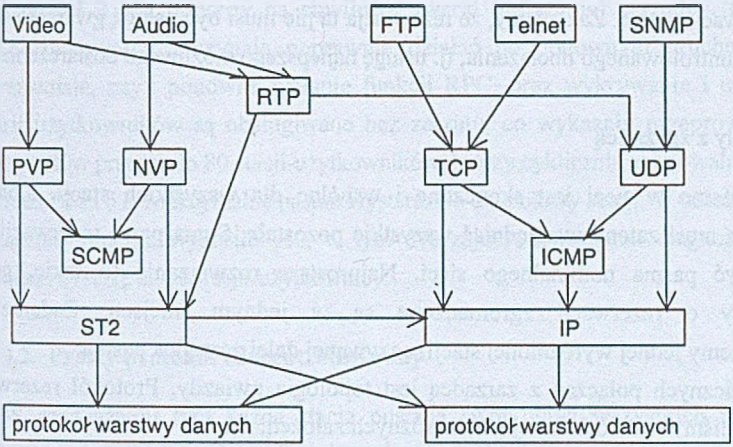
ST2 jest eksperymentalnym protokołem w Internecie i jest odpowiednikiem protokołu IP. ST2 [4] [5] dostarcza aplikacjom mechanizmów rezerwacji zasobów w sieci. Razem z odpowiednim szeregowaniem pakietów w routerach, ST2 może zapewniać odpowiednią jakość połączenia w Internecie. ST2, podobnie jak IP, składa się z dwóch protokołów:

1. ST - do transportu danych oraz
2. SCMP (Stream Control Message Protocol) - do sterowania.

ST2 jest odpowiednikiem IP w warstwie sieciowej, patrz rysunek 1.

Główną różnicą jest to, że ST2 dostarcza połączeniowego transferu danych w czasie rzeczywistym, a IP bezpołączeniową usługę najlepszego możliwego dostarczenia. Tak jak TCP i UDP są protokołami warstwy transportowej współdziałającymi z IP, tak RTP (Real Time Protocol), PVP (Packet Video Protocol) i NVP (Network Voice Protocol) są protokołami

warstwy transportowej bazującymi na ST2. Nagłówki ST2 i IP różnią się polem numeru wersji (numery te to 4 lub 6 dla IP, i 5 dla ST2).



Rys. 1. ST2 jako odpowiednik IP
Fig. 1. ST2 as a counterpart of IP

RTP może używać UDP/IP w celu najlepszego możliwego dostarczenia ruchu czasu rzeczywistego. Również RTP może korzystać z ST2 w celu dostarczenia ruchu czasu rzeczywistego z zagwarantowaną jakością usługi. Możliwa jest również enkapsulacja pakietów ST2 w pakietach IP.

3. Proponowane protokoły rezerwacji dla popularnych sieci LAN

Powyższej omawiane protokoły nie są stosowalne w sieci LAN. Definiują one głównie komunikaty i synchronizację pomiędzy ruterami w sieci rozległej. W sieci lokalnej nie występuje problem synchronizacji ruterów, lecz problem dzielenia wspólnego medium, jakim jest np. jeden segmentu Ethernetu.

Proponowane niżej protokoły nie mają na celu zagwarantowania bezkolizyjnej pracy. Nie dostarczają one usługi gwarantowanej, lecz mechanizm rezerwacji udostępniany aplikacjom. Mechanizm będzie działał poprawnie, pod warunkiem że nikt nie będzie przekraczał deklarowanego obciążenia oraz że przewidywalne jest maksymalne obciążenie segmentu przez inne aplikacje nie korzystające z tego mechanizmu. Aczkolwiek możliwe jest wprowadzenie mechanizmu egzekwowania zadeklarowanych przez aplikacje pasm na poziomie systemu operacyjnego, w tym artykule nie rozpatrujemy tego zagadnienia.

Przez popularną sieć LAN rozumiemy sieć Ethernet, Token Ring lub inną sieć nie oferującą usługi rezerwacji pasma. Dyskutowane niżej rozwiązania mają wzbogacić taką sieć o mechanizm rezerwacji pasma. Zakładamy, że rezerwacja ta nie musi być usługą gwarantowaną, a jedynie usługą kontrolowanego obciążenia, tj. usługę najlepszego możliwego dostarczenia.

3.1. Protokoły z zarządcą

Nominalne pasmo w sieci jest skończone i wspólne dla wszystkich stacji. Kolejna rezerwacja pasma musi zatem uwzględniać wszystkie pozostałe. Suma pasm rezerwacji nie może przekroczyć pasma nominalnego sieci. Najprostsze rozwiązanie to takie, gdzie wszystkie zapisy o rezerwacji zgromadzone są w jednym miejscu. Zadanie to przyporządkowujemy jednej wyróżnionej stacji, nazywanej dalej zarządca pasma.

Strukturą logicznych połączeń z zarządcą jest topologia gwiazdy. Protokół rezerwacji pasma zrealizowaliśmy w dwóch wersjach dla różnych założeń:

1. zarządca jest niezawodny, komunikacja niezawodna, stacje niezawodne,
2. zarządca jest niezawodny, komunikacja zawodna, stacje zawodne.

W protokole 2 uwzględniającym zawodność komunikacji i możliwość awarii stacji zarządca przechowuje szczegółowe informacje o każdej rezerwacji, które umożliwiają wykrywanie sytuacji, takich jak np. uszkodzona stacja nigdy nie zwolni pasma. Każda rezerwacja oprócz wielkości pasma znakowana jest czasem, odpowiednikiem daty ważności lub przydatności do spożycia. Dlatego stacje sprawne zobligowane są do cyklicznego raportowania o zajmowanym paśmie. Jeżeli nastąpi awaria, to stacja przestanie wysyłać raporty, co spowoduje, że zarządca anuluje rezerwację.

Jeżeli zaginie komunikat zamawiający pasmo przez stację, to nie otrzyma ona odpowiedzi. Spowoduje to retransmisję. Z kolei retransmisje zwiększają prawdopodobieństwo występowania duplikatów komunikatów w sieci, co jest wykrywane przez zarządcę. Jeżeli przyjdzie duplikat komunikatu rezerwującego, to sprawdzane jest najpierw, czy pasmo zostało już wcześniej zarezerwowane przez tę stację. Nie dochodzi do sytuacji przydziału większej ilości pasma,niż żąda tego użytkownik.

Do komunikacji między zarządcą i użytkownikiem stosowane jest zdalne wywołanie procedur (RPC). Obydwa protokoły zaimplementowaliśmy w języku C.

Zamówienie pasma polega na wywołaniu przez stację zdalnej procedury na komputerze zarządcy. Procedura ta nazywa się *rezerwuj*. Do zwalniania pasma służy procedura *anuluj*. Trzecia i zarazem ostatnia zdalna funkcja to *pytaj*. Zwraca ona wielkość pasma użytkownika i jednocześnie przedłuża ważność rezerwacji. Przed użyciem tych funkcji aplikacja musi wywołać funkcję *rpc_ini*, a po użyciu funkcję *rpc_end*.

Programy zarządców obu protokołów 1 i 2 różnią się tym, że pierwszy z nich nie zapamiętuje, komu przydzielił pasmo. Dlatego nie jest odporny na awarie stacji.

Protokół 2 jest odporny na chwilowe awarie (milczenie) zarządcy, tzn. uruchomione stacje-użytkownicy zaczynają poprawnie działać po ponownym uruchomieniu systemu. Retransmisje, czyli ponowne wołanie funkcji RPC, oraz wykrywanie i usuwanie skutków awarii użytkowników są obsługiwane bez zarzutu, co wykazały przeprowadzone testy. W czasie testów pracowało 80 stacji-użytkowników, którzy cyklicznie zajmowali i zwalniali pasmo na okres 10 s i 5 s. Maksymalne pasmo wystarczało dla połowy stacji. W czasie testu protokołu 2 obciążenie systemu wynosiło 9%, w tym 8% zajmował program diagnostyczny *top*, 0.1% zarządca, a resztę 0.9% stacje-użytkownicy.

3.2. Prosty protokół zdecentralizowany

W rozwiązaniu tym każda stacja posiada swój obraz wszystkich rezerwacji. Na tej podstawie może przyjąć lub odrzucić żądanie użytkownika o zarezerwowanie pasma. Jeżeli kolejna rezerwacja spowodowałaby przekroczenie nominalnej przepustowości sieci, to jest odrzucana, a użytkownik otrzymuje sygnał zajętości. Jeżeli żądane pasmo jest wolne, stacja przyjmuje podanie oraz modyfikuje swój stan rezerwacji.

Algorytm działa tylko wtedy, gdy stacje mają taki sam obraz dopuszczonych rezerwacji. Przyjęliśmy, że stacje należą do jednego segmentu sieci. Komunikat protokołu rezerwacji odbierają wszyscy członkowie. Po zarezerwowaniu pasma stacja wysyła komunikat zawierający informację o nowym stanie wszystkich rezerwacji. Zakładamy, że protokół obsługuje jeden segment sieci i jednocześnie może być nadany jeden komunikat. Pozostałe stacje odbierają komunikat przydziału rezerwacji i będą aktualizowały swoje stany rezerwacji. Jeżeli stacja po uruchomieniu chce dowiedzieć się o zarezerwowanym paśmie i przyspieszyć wysłanie przez kogoś raportu o zarezerwowanym paśmie, to wysyła komunikat z pytaniem i czeka na odpowiedź. Jeżeli wszystkie stacje odpowiadałyby jednocześnie na pytanie, to powodowałoby to zbędny ruch w sieci. Dlatego też stacje opóźniają o losowy czas odpowiedź, a jeżeli odbiorą ją od innej stacji w tym czasie, to anulują swoją. W ten sposób wysłany zostanie jeden raport.

Pierwsza stacja włączająca się do sieci wysyła komunikat zapytania, ale nie otrzyma od nikogo odpowiedzi. Po kilku bezskutecznych retransmisjach pakietu stacja przyjmuje, że jest jedyna i przydziela sobie całe pasmo nominalne sieci.

Każda stacja wykonuje ten sam algorytm. Algorytm ten zaimplementowaliśmy jako protokół 3. Zrealizowaliśmy w nim pewne optymalizacje zwiększające jego niezawodność. Przy przyjętych dwóch założeniach (o uporządkowaniu pakietów) oraz braku gubienia komunikatów nie występuje problem współzawodnictwa. Dzieje się tak, ponieważ tylko

jedna z dwóch stacji, chcących zarezerwować sobie pasmo w tym samym czasie, może wysłać komunikat żądania. Druga stacja odbierze ten komunikat i zweryfikuje swoją rezerwację (ewentualnie do niej nie dopuści). Ale, jak przystosować ten protokół do sieci, która nie spełnia tych założeń.

Jeżeli protokół ten implementować będziemy w warstwie IP, to przekazanie komunikatu do warstwy MAC jest jego wysłaniem. Globalnie w całym segmencie sieci może więc być wysłanych jednocześnie wiele pakietów żądania pasma. W efekcie gubienia pakietów w medium i ich repetycji przez protokół warstwy MAC może dojść do zmiany uporządkowania.

Problem z uporządkowaniem komunikatów można rozwiązać przez znakowanie czasem wysłania przesyłanych pakietów. Wtedy stacje odbierające mogą uporządkować odbierane pakiety względem czasu ich nadania. Można określić dla sieci pewien maksymalny czas graniczny pomiędzy nadaniem a odebraniem pakietu. Właśnie w tym czasie dokonujemy porządkowania odebranych pakietów. Na ten czas stacja pozostaje w stanie OCZEKUJE po wysłaniu do sieci komunikatu REZERWUJ. Jeżeli okaże się, że rezerwacja stacji A wysłana została po innej rezerwacji od stacji B i pakiet z B nie został odebrany w stacji A przed wysłaniem własnego, to ta rezerwacja z A jest odrzucana i wtedy konstruowany jest nowy komunikat Rezerwuj.

Aby uodpornić protokół na ginięcie komunikatów, można zobligować stacje do potwierdzania każdej rezerwacji. Ale potwierdzanie przez wszystkie stacje byłoby zbyt kosztowne. Wystarczy potwierdzenie od jednej losowo wybranej stacji. Wybór stacji może być następujący. Wszystkie stacje inicjują czasomierz wartością losową z pewnego przedziału ograniczonego od góry zadaną stałą. Po upłynięciu czasu danej stacji wysyła ona potwierdzenie, a inne stacje anulują czasomierz i nie wysyłają żadnego potwierdzenia. W ten sposób wysłane zostanie tylko jedno potwierdzenie, które jest bardzo odporne na zaginięcie, tzn. prawdopodobieństwo zaginięcia tego pakietu wynosi p^n, gdzie p - prawdopodobieństwo zaginięcia jednego komunikatu POTWIERDZENIE , n - liczba stacji .

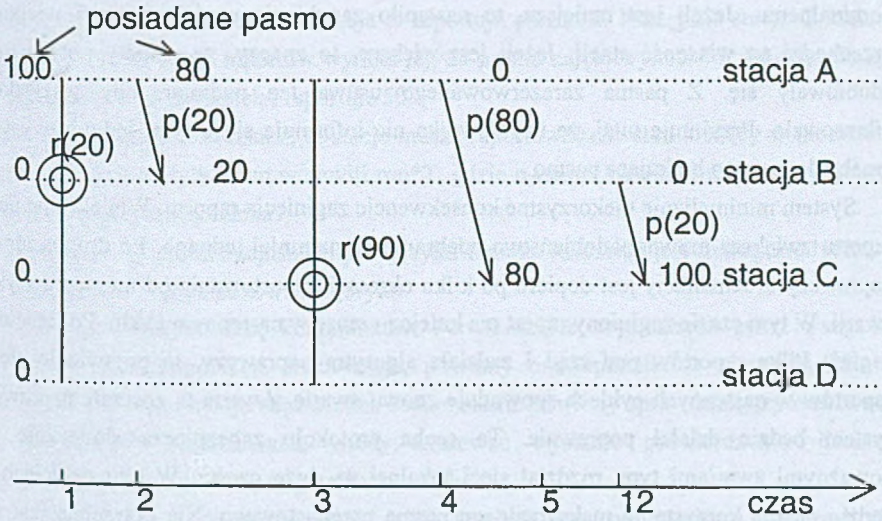
Modyfikacją powyższej metody jest to, aby stacja musiała jednak zgromadzić pewną ilość potwierdzeń, np. 50%, a stacje wysyłają potwierdzenia niezależnie od innych. Wtedy wszystkie stacje przeprowadzają coś w rodzaju wyborów demokratycznych bezpośrednich i ogół stacji decyduje o rezerwacji innej stacji.

3.3. Protokół zdecentralizowany o podwyższonej niezawodności

Algorytm stacji jest modyfikacją algorytmu zajmowania zasobów Ricarta-Agrawali(1981). Modyfikacja polega na zastąpieniu potwierdzenia typu binarnego potwierdzeniem stanowiącym liczbę reprezentującą wielkość pasma, jakie odbiorca może wykorzystywać. Stacja, która otrzyma od innych zezwolenia na korzystanie z pasma o

zamawianej wielkości, powiadamia aplikację i rozpoczyna się transmisja. Protokół jest odporny na awarię dowolnego podzbioru stacji, oraz na ginięcie lub dublowanie pakietów. Struktura komunikacji jest taka sama jak w protokole 3, tzn. stacja sama decyduje o przyjęciu rezerwacji oraz występuje komunikacja rozgłoszeniowa.

Automat protokołu jest trójstanowy. Stanem początkowym jest INI. Wtedy, po odebraniu komunikatu Rezerwacja, stacja albo wysyła potwierdzenie przekazania pasma, gdy dysponuje wolnym pasmem, albo wprowadza tę rezerwację do kolejki FIFO. Kolejka ta zawiera rezerwacje, które będą zrealizowane, gdy stacja będzie dysponowała wolnym pasmem, np. po zwolnieniu przez użytkownika. Załóżmy, że użytkownik zdecydował się zarezerwować pasmo. Stacja przechodzi do stanu OCZEKUJE i wysyła do grupy komunikat Rezerwacja. W stanie tym nie są obsługiwane przychodzące komunikaty Rezerwacja. Są one zapisywane w kolejce do późniejszej obsługi.7 Stacja czeka na komunikat Potwierdzenie, który oznacza przekazanie części pasma nadawcy do dyspozycji odbiorcy tego komunikatu. Gdy stacja nazbiera odpowiednio wielkie pasmo większe lub równe zadanemu, powiadamia o tym użytkownika i przechodzi ze stanu OCZEKUJE do stanu ZAREZERWOWANE. Nadmiarowe pasmo, które może otrzymać stacja, zostaje rozdawane przy przetwarzaniu komunikatu Rezerwacja. Gdy użytkownik zwalnia pasmo, wówczas stacja przechodzi do stanu INI i cykl się zamyka. Działanie algorytmu ilustruje rysunek.



Rys. 2. Przykład działania protokołu 4

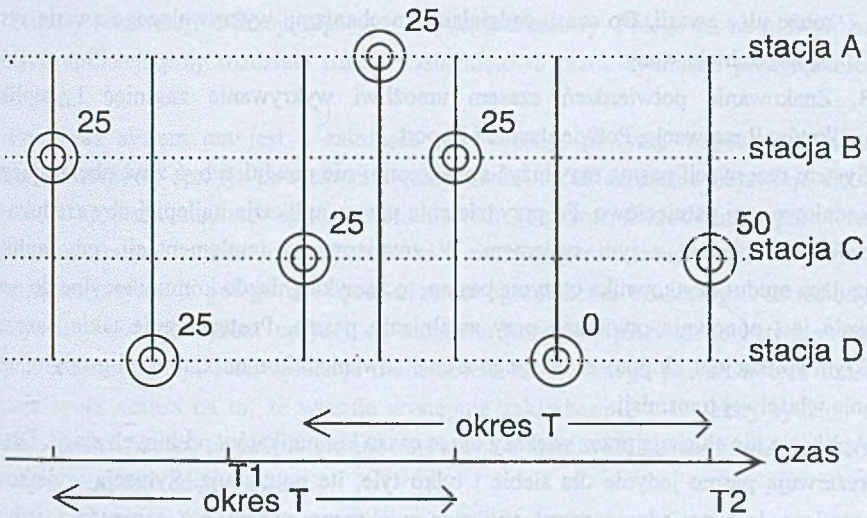
Fig. 2. An example of protocol 4 operation

Rysunek przedstawia dwie rezerwacje pasma. Stacja B rezerwuje pasmo 20 jednostek na czas 10 , a stacja C 90 jednostek. Nominalna przepustowość sieci wynosi 100 jednostek.

Widać więc, że te dwie rezerwacje nie mogą wystąpić równocześnie. Zobaczymy, że druga rezerwacja zostanie wstrzymana do czasu zwolnienia pasma przez pierwszą. Na początku całym pasmem dysponuje stacja A. W chwili t=1 stacja B rozgłasza komunikat r=REZERWUJ z prośbą o pasmo 20. Stacja A nie rezerwuje sobie pasma, więc może wysłać p=Potwierdzenie przekazania pasma 20. Stacja A ma teraz 80, a B 20 jednostek. Widać, że stacja B zarezerwowała sobie odpowiednie pasmo. Do czasu anulowania rezerwacji przez użytkownika stacja będzie trzymała to pasmo. Następnie w chwili t=3 stacja C zgłasza zapotrzebowanie na pasmo 90. Stacja A nadal nie rezerwuje sobie pasma, więc przesyła to co ma, tzn. 80. Ale brakuje jeszcze 10 jednostek. Stacja B zwolni swoje pasmo w chwili t=12. Po tym przesyła komunikat potwierdzający do stacji C. Widać tutaj, jak algorytm nie dopuszcza do zajęcia dwukrotnego tego samego zasobu oraz że stacja niezainteresowana rezerwacjami nie musi wysyłać jakichkolwiek komunikatów (np. stacja D).

Aby system był odporny na awarie, stacje wysyłają okresowo raport o zajętych przez siebie zasobach. Raporty umożliwiają odtworzenie informacji o wolnym paśmie, które wcześniej zajmowała uszkodzona teraz stacja. Działanie tego mechanizmu jest następujące. Okres zaczyna się po wysłaniu raportu. Od tej pory stacja sumuje zadeklarowane pasmo w raportach nadchodzących od innych stacji. Pod koniec okresu, tuż przed wysłaniem własnego raportu, stacja sprawdza, czy ta suma razem z własnym pasmem równa jest pasmu nominalnemu. Jeżeli jest mniejsza, to nastąpiło zagubienie części pasma i wielkość ta przechodzi na własność stacji. Jeżeli jest większa, to znaczy, że pakiety potwierdzające zdublowały się. Z pasma zarezerwowanego usuwa ten nadmiar, aby wszystko się bilansowało. Przyjmuję tutaj, że użytkownika nie informuje się o tym, jedynie wysyła się prośbę do grupy o brakujące pasmo.

System minimalizuje niekorzystne konsekwencje zaginięcia raportu. Wielokrotne nadanie raportu zwiększa prawdopodobieństwo odebranie co najmniej jednego. Po drugie, algorytm naprawczy uruchamiany jest dopiero po kilku okresach raportowania od momentu wykrycia awarii. W tym czasie zaginiony raport ma kolejną szansę w następnym cyklu. Po trzecie, jeśli zaginie kilka raportów pod rząd i zadziała algorytm naprawczy, to pojawienie się tych raportów w następnych cyklach spowoduje znowu awarię. Awaria ta zostanie naprawiona i system będzie działał poprawnie. Ta cecha protokołu zabezpiecza skutecznie przed poważnymi awariami typu rozdział sieci lokalnej na dwie części. W obu częściach sieci będzie można korzystać z maksymalnego pasma przepustowego. Nie potrzebna jest nawet interwencja administratora. Przykład działania tego algorytmu ilustruje rysunek 3.



Rys. 3. Wykrywanie awarii w protokole 4
Fig. 3. Failure detection at protocol 4

Stacje rozgłaszają z okresem T raporty o posiadanym paśmie. Początkowo wszyscy mają po 25 jednostek. W chwili T1 stacja D wysłała pakiet potwierdzający pasmo 25, który zaginął. Zgodnie ze swoim stanem stacja D raportuje pasmo 0. Następnie stacja C kończy swój okres i z sumowania raportów wynika jej, że pasmo zajęte wynosi 75. Dlatego dodaje sobie brakujących 25 jednostek i raportuje 50.

Aby ten mechanizm zadziałał, to stacje muszą raportować nie stan bieżący w momencie generowania raportu, tylko stan w chwili t=n*T, gdzie n-numer okresu, lub też stacje muszą śledzić wszystkie potwierdzenia.

Algorytm w tej postaci zapisaliśmy w języku Estelle. Możliwy jest szereg modyfikacji, które mają na celu zoptymalizować algorytm.

1. Aby zmniejszyć liczbę komunikatów Potwierdzenie, wysyłanych przez wszystkie stacje po komunikacie Rezerwacja, powinny one opóźniać o niewielki losowy skończony czas swoje odpowiedzi. Jednocześnie powinny sprawdzać, czy rezerwacja już została zrealizowana, wtedy anulować wysłanie potwierdzenia. Efektem ubocznym jest większa fragmentacja zasobu.
2. Fragmentacja wolnego pasma na wielu stacjach jest kompromisem między liczbą komunikatów a niezawodnością sieci. Im większa fragmentacja, tym zasoby są równomiernie rozłożone, wtedy awaria jednej stacji mało wpływa na cały system. Wadą dużej fragmentacji jest większa liczba komunikatów potwierdzeń od wielu stacji. Z kolei mała fragmentacja oznacza skupienie zasobów w jednej stacji, która

może ulec awarii. Do czasu zadziałania mechanizmu wykrywającego awarie system będzie zablokowany.

3. Znakowanie potwierdzeń czasem umożliwi wykrywanie zaginięć i duplikacji komunikatów Rezerwacja, Potwierdzenie i Raport.

System rezerwacji pasma ma służyć aplikacjom i nie powinien być zbyt obciążający ani obliczeniowo, ani pamięciowo. Po przydzieleniu pasma aplikacja najlepiej aby zredukowała do zera komunikację z tym systemem. W zrealizowanej implementacji, gdy aplikacja realizująca moduł użytkownika otrzyma pasmo, to zamyka gniazdo komunikacyjne do grupy. Gniazdo jest ponownie otwierane przy zwalnianiu pasma. Postępowanie takie oszczędza zwykłym aplikacjom kłopotu związanego z analizowaniem komunikatów od grupy w czasie trwania właściwej transmisji.

Aplikacje nie słuchają przez większy okres czasu komunikatów od innych stacji. Dlatego też rezerwują pasmo jedynie dla siebie i tylko tyle, ile potrzebują. Sytuacją wyjątkową i niepożądaną jest posiadanie przez aplikację większego pasma niż zamawiała, gdyż ta nadwyżka nie może być wykorzystana przez nikogo.

Aplikacje ulegają „awariom", niekoniecznie związanymi z uszkodzeniami sprzętu. Mogą być awaryjnie przerwane przez administratora lub użytkownika. Konsekwencją takiego „uszkodzenia" jest często niezwolnienie zarezerwowanego pasma. Aby zabezpieczyć system rezerwacji przed taką okolicznością, przydział pasma jest aktualny w określonym czasie. Aby przedłużyć rezerwację, stacja musi udowodnić, że jeszcze istnieje. Robi to przez wysłanie komunikatu.

Drugą grupę stacji tworzą procesy dedykowane systemowi rezerwacji pasma. Są one zarządzane przez administratora i nie służą bezpośrednio zwykłym użytkownikom lub aplikacjom. Przeważnie procesy te działają przez cały czas działania systemu rezerwacji. Ewentualne przerwy są związane z uszkodzeniem sprzętu lub też świadomą decyzją administratora o zakończeniu działania.

Procesów rezerwacji może być wiele. Nie muszą one być uruchomione na wszystkich stacjach danej sieci. Może się zdarzyć, że kilka z nich będzie gotowych zrealizować pojawiające się zamówienie. Niewłaściwym działaniem byłoby przekazanie przez wszystkie te stacje pasma, ponieważ aplikacja otrzymałaby w sumie wielokrotność zamówienia, a nie jednokrotność. Dlatego też stacje dedykowane opóźniają o losowy czas T swoją odpowiedź, albo nawet ją anulują, jeśli usłyszą, że zamówienie zostało zrealizowane. Czas T jest losowany z zakresu (T1,T2), np. T1=200ms, T2=5 s, wedle rozkładu jednostajnego. Taki mechanizm opóźniania potwierdzenia wydłuża czas wykonania rezerwacji do minimum T1. Aby tego uniknąć, wprowadzona jest następująca modyfikacja. Aplikacja może wyróżnić w swoim zamówieniu jedną stację dedykowaną, na której pracuje proces rezerwacji. Jeśli tylko nie uległa ona awarii, odpowiada bez zwłoki.

Procesy rezerwacji realizują algorytm antyzagłodzeniowy. Polega on na niewyróżnianiu żadnych aplikacji przy rozdziale pasma. Osiągnięto to przez cykliczne przeglądanie listy zamówień.

Ponieważ system ten jest z założenia rozproszony, powyżej wspomniany algorytm antyzagłodzeniowy sprzyja powstawaniu zakleszczeń[1] . Zakleszczenia objawiają się w ten sposób, że aplikacja nie może zgromadzić zamawianego pasma, tzn. otrzymała potwierdzenia przekazania pasma o sumarycznej wielkości mniejszej niż zamówienie. Wykrycie tej niepożądanej sytuacji polega na wprowadzeniu ograniczenia czasowego. Określony jest maksymalny czas, jaki może upłynąć od momentu przybycia pierwszego potwierdzenia do zaspokojenia aplikacji. Jeśli aplikacja nie nazbiera zamawianego pasma w tym czasie, to istnieje spora szansa na to, że właśnie występuje zakleszczenie[2]. W takiej sytuacji stacja zwalnia całe posiadane pasmo, informuje inne stacje o możliwości wystąpienia zakleszczenia i zaczyna wszystko od początku.

Jeśli zakleszczenie zostało już wykryte, to stacje dedykowane próbują je usunąć. Anulują algorytm antyzagłodzeniowy i nie respektują wyróżnienia stacji dedykowanej w zamówieniach aplikacji. W zamian stosują algorytm antyzakleszczeniowy, który polega na wysłaniu potwierdzenia pasma tej stacji, której najmniej brakuje.

Innym problemem są awarie. Jak z powyższego wynika, awarie użytkowników nie są groźne, ponieważ po okresie raportowania uszkodzona stacja nie wyśle raportu i w ten sposób demon wykreśli tę rezerwację ze swojej tabeli. Trudniejsze jest natomiast przeciwdziałanie awariom demonów.

Powielanie pakietów jest łatwo wykrywalne poprzez dodanie mechanizmu ich identyfikacji (numer seryjny lub czas nadania). Większe problemy są powodowane przez zaginięcie pakietu, np. Anuluj_Pasmo. Pasmo będzie niezwolnione, aż do czasu upłynięcia okresu raportowania. Wtedy algorytm wykrywania i usuwania awarii spowoduje, że system odzyska „zagubione pasmo".

Największe kłopoty sprawiają operacje nieidempotentne. Chodzi tu o komunikaty Zamówienie, oraz Przekazanie_Pasma, w szczególności przekazanie pasma innemu demonowi 12. Zastosowany algorytm wykrywający i odrzucający duplikaty polega na znakowaniu czasem komunikatu przez nadawcę. Procesy rezerwacji pamiętają czas nadania ostatniego pakietu od innych stacji. Pakiety mające czas nadania starszy lub równy uznawane

[1] Zakleszczenia takie powstają najczęściej, gdy dwie stacje ubiegają się jednocześnie o pasmo bliskie przepustowości maksymalnej oraz wyróżniają dwie stacje dedykowane, które posiadają równe ilości pasma, w sumie tyle ile potrzeba dla jednej rezerwacji. Ilustracją tej sytuacji jest test zawarty w [2].

[2] Inną przyczyną może być po prostu brak wolnego pasma.

są za duplikaty. Stosowany w implementacji zegar generuje impulsy co 1 mikrosekundę. Dlatego w ciągu 1 μs może być nadany tylko jeden komunikat przez określony proces rezerwacji.

W celu zmniejszenia awaryjności systemu związanej z utratą pakietów można nakazać, aby stacje od razu nadawały n kopii każdego pakietu. Jeśli założyć, że pakiet ginie z prawdopodobieństwem p, to wszystkich n pakietów zaginie z prawdopodobieństwem p^n. Liczba n powinna być dobierana w zależności od p i oczekiwanej awaryjności systemu.

3.4. Protokół o strukturze pierścienia logicznego

Dla działania tego protokołu stacje muszą być zorganizowane w pierścień. Jeżeli nie jest to fizyczny pierścień (np. Token Ring), to konieczna jest budowa pierścienia logicznego (np. Token Bus). Każda stacja ma swojego poprzednika i następnika. W pierścieniu krążą znaczniki, którym przyporządkowujemy wolne pasmo. Jeżeli stacja chce zarezerwować pasmo, to czeka ona na znacznik z odpowiednią częścią pasma i przechwytuje go na czas rezerwacji. Znacznik może być podzielony na dwa. Jeden z nich o wielkości równej zamawianemu pasmu zostaje w stacji, a drugi, będący resztą, zostaje przekazany następnikowi. Zwolnienie pasma polega na przekazaniu znacznika do swojego następnika. Jeżeli stacja nie jest zainteresowana rezerwacjami, to po prostu przekazuje znacznik następnikowi.

W sieciach pierścieniowych wybierana jest (zwykle dynamicznie) stacja monitora realizująca szereg funkcji diagnostycznych i serwisowych. Stacja ta musi zostać obciążona dodatkowymi czynnościami związanymi z obsługą znaczników rezerwacji pasma, tj.:

- odzyskiwanie znaczników zawłaszczonych przez uszkodzoną stację,
- odzyskiwanie znaczników zagubionych w medium transmisyjnym.

Efektywność tego protokołu będzie największa, jeżeli zostanie on wkomponowany w protokół warstwy MAC sieci pierścieniowej. Z tego powodu dla różnych sieci lokalnych realizacja tego protokołu będzie przebiegać zupełnie inaczej. Implementacja tego protokołu w sieci Ethernet z poziomu protokołu UDP czy nawet IP nie wydaje nam się opłacalna ze względów wydajnościowych. Toteż nie był on przez nas ani testowany, ani implementowany.

4. Podsumowanie

W wielu sieciach występują serwery plików służące stacjom roboczym. Awaria serwera powoduje praktycznie unieruchomienie wszystkich komputerów. W takich sieciach dobrze

pracuje protokół z zarządcą (protokół 2), który również jest zależny od awarii jednego centralnego elementu. Stosowanie tego protokołu nie zwiększa poziomu awaryjności systemu.

Protokół 4 jest najbardziej odporny na awarie sieci i komputerów. W początkowych fazach działania protokołu mogą powstawać chwilowe zakleszczenia, ale są one wykrywane i usuwane. Później już nie występują. Zaletą tego protokołu są niewielkie nakłady na administrowanie systemem rezerwacji pasma. Nie trzeba konfigurować każdej aplikacji do korzystania z wyróżnionego zarządcy, nie trzeba podawać jego adresu i innych parametrów. Zasięg systemu rezerwacji jest ograniczony dokładnie do jednego segmentu sieci. Protokół ten wymaga niewiele większych nakładów obliczeniowych niż protokół 2.

Protokół 1 powstał w celu porównania z bardzo podobnym protokołem 2. Różnicą jest brak odporności protokołu 1 na awarie sieci i komputerów. Z obserwacji wynika, że protokoły 1 i 2 działają tak samo sprawnie. Protokół 1 można stosować dla niekrytycznych aplikacji.

Niektóre systemy operacyjne, przykładem jest Linux 2.2[1], dostarczają mechanizmy egzekwowania ustalonych statycznie poziomów przepustowości strumieni danych generowanych przez aplikację. Systemy takie mają rozbudowany proces szeregujący pakiety do nadania, który zawiera wiele kolejek priorytetowych. Nowym elementem jest filtr pakietów, który rozdziela pakiety do różnych kolejek. Jeżeli zastosujemy w stosunku do strumienia danych filtr typu TBF (Token Bucket Filter) [5] [6] [7], to system operacyjny będzie egzekwował założoną wcześniej przepustowość. Rozdział pasma odbywa się statycznie, tzn. w plikach konfiguracyjnych, które wczytywane są przy starcie systemu. Zapisane tam pasmo otrzymują usługi: telnet, ftp, mail lub strumień do konkretnego adresata. Sprzężenie systemu rezerwacji pasma z takim mechanizmem kontroli przepływu umożliwiłoby dynamiczne sterowanie filtrem i procesem szeregującym, np. poprzez wywołania funkcji systemowych.

Kolejnym krokiem powinno być jeszcze ściślejsze powiązanie systemu rezerwacji pasma z system operacyjnym. Ideałem byłoby, aby aplikacje rezerwowały pasmo poprzez wywołanie funkcji systemowej typu *setsockopt*, która służy do ustalania różnych parametrów gniazda BSD. Poprzez zadanie nowego parametru tej funkcji można by rezerwować pasmo dla tego strumienia. Przystosowanie starych aplikacji do systemu rezerwacji pasma byłoby bardzo proste i polegałoby jedynie na dodaniu wywołania takiej funkcji systemowej.

Odrębnym trudnym i ciekawym tematem przyszłych prac jest opracowanie współpracy lokalnego protokołu rezerwacji pasma z protokołem stosowanym w sieci globalnej, takim jak RSVP lub ST2.

---

[1] Opis mechanizmu znajduje się w pakiecie oprogramowania *iproute2* oraz na stronach *man tc* i *man ip*

LITERATURA

1. Estelle Development Toolset (EDT), http://www-lor.int-evry.fr/edt/.
2. Żółkowski C.: Protokoły rezerwacji pasma przepustowości w sieciach TCP/IP. Praca Inżynierska, Instytut Informatyki PW, marzec 2000, http://www.ii.pw.edu.pl/~czolkows/pd.
3. RFC2205 - Resource ReSerVation Protocol (RSVP).
4. ST2 Protocol - RFC1190, RFC1819, RFC1946, RFC2383.
5. Sanjoy P.: Multicasting on the Internet and its applications. Kluwer Academic Publisher, 1998.
6. Kuznetsov A.: Oprogramowanie i dokumentacja iproute2, ftp://ftp.inr.ac.ru/ip-routing/.
7. Krawczyk P.: Sterowanie przepływem danych w Linuxie 2.2, www.ceti.com.pl/~kravietz/cbq.

Recenzent: Dr inż. Bartłomiej Zieliński

Wpłynęło do Redakcji 3 kwietnia 2000.

Abstract

Since several years we observe a rapid development of computer networks and services they offer. Some of these services need a bandwidth reservation facility to guarantee a demanded by user quality, e.g., voice/image exchange, time depended control systems. However most of currently used LANs do not offer the bandwidth reservation facility. Users that want the new services are obliged to change their Ethernet or Token Ring equipment to e.g., ATM equipment, which could be too expensive for some of them. Is it possible to enhance the commonly used LANs, that they offer the bandwidth reservation for applications, which need it?

To solve this problem we have defined five protocols:

1. Protocol with a simple manger - a selected computer that governs the bandwidth. The protocol is dedicated for a reliable network.
2. Reliable protocol with a manger. The network cans loss messages and computers can fail.
3. Decentralized protocol for a reliable network.
4. Reliable decentralized protocol that is based on a modified Ricart Agrawal algorithm.

5. Reliable decentralized protocol that works on a token ring base.

We assume that all protocols work at one segment LAN. The applications that need bandwidth reservation facility use TCP/IP protocols. Our protocols can be implemented under UDP on the beginning, and after inside IP level. Moreover we assume that the network load is predicable for all applications, both for these that use the reservation facility and for those that know about it.

We have specified all the five protocols in Estelle language in order to build a precise documentation and to analyse them. We have implemented in C protocols 1, 2, 4, and we have tested them. This paper is a kind of recapitulation of these experiments. On the beginning we discus the bandwidth reservation protocols dedicated for WANs, namely RSVP and ST2. After we describe our protocols. All of them can be applied in a real system depending on the user needs and the LAN properties.

The future enhancements of the protocols can lead to their integration with an operating system. This integration allows creation of a control mechanism that will guarantee the application respects the reservation agreements.