ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1998
Seria: INFORMATYKA z. 34 Nr kol. 1381

Mirosław SKRZEWSKI, Piotr KASPRZYK, Adam DOMAŃSKI
Politechnika Śląska, Instytut Informatyki

ORGANIZACJA MONITOROWANIA RUCHU PAKIETÓW W SIECI, PROBLEMY KONSTRUKCJI AGENTA PROTOKOŁU SLMP

Streszczenie. W pracy przedstawiono propozycję „rozproszonego" systemu analizy ruchu pakietów w sieci lokalnej, umożliwiającego określenie dokładnych zależności czasowych opisujących ruch pakietów w sieci wielosegmentowej. Przedstawiono ogólną koncepcję organizacji rejestracji ruchu pakietów, wstępnego przetwarzania / kompresji wyników pomiarów oraz omówiono wybrane zagadnienia konstrukcji modelu agenta pomiarowego stacji sieci.

SELECTED PROBLEMS OF NETWORK PACKET FLOW MONITORING, THE SLMP AGENT IMPLEMENTATION ISSUES

Summary. Proposition of the distributed network packet flow monitoring system with the possibility of determining exact packet flow timing relationships has been described. General concept of packet flow monitoring, data compression and acquisition, time measurement are presented. Technical problems of the protocol agent implementation and some early experiment results are also described.

1. Narzędzia badania ruchu pakietów, ich możliwości i ograniczenia

Badanie ruchu pakietów w sieci możliwe jest za pomocą dedykowanych do tego urządzeń - sprzętowych analizatorów protokołów lub za pomocą specjalnych programów wykorzystujących możliwości standardowych kart sieciowych komputerów - analizatorów software'owych. Narzędzia te podłączone są do konkretnego segmentu sieci. Ich praca polega na odbiorze wszystkich pakietów przesyłanych w danym segmencie i ich analizie.

Takie rozwiązania dają wgląd tylko w działanie fragmentu sieci, do którego są dołączone. Informacje o dalszym otoczeniu sieciowym wynikają jedynie z ewentualnego ruchu zewnętrznych pakietów poprzez dany segment.

Możliwości tego typu narzędzi sprowadzają się do obserwacji różnych charakterystyk sieci mniej lub bardziej szczegółowych, np. dla sieci Ethernet można przykładowo monitorować następujące elementy:

- liczby ramek na sekundę, liczby bitów na sekundę, ilość błędnych ramek, średni czas między ramkami,
- listę komputerów wysyłających i odbierających najwięcej ramek,
- listę wykorzystywanych protokołów.

W zależności od typu protokołu można dowiedzieć się również o bardziej szczegółowych danych, np. dla sieci TCP/IP można obserwować:

- zawartość tablicy ARP (Address Resolution Protocol, RFC 826);
- który komputer wysyła nierozwiązywalne zapytania ARP;
- który komputer przesyła najwięcej ramek w protokołach IP, UDP i TCP;
- jakie rodzaje protokołów są używane;
- który komputer jest najbardziej wykorzystany jako serwer/klient protokołu TELNET;
- kolejność ramek w protokołach BOOTP i TFTP;
- charakterystykę danego komputera - rozmiar fragmentów, maksymalna wielkość ramki (Maximum Transmission Unit - MTU), opcje nagłówka IP;
- zdarzenia i nazwy protokołu NetBIOS (według RFC 1001 i 1002);
- pracę protokołu DNS;
- ważne zdarzenia protokołu TCP, takie jak rozpoczęcie i zakończenie połączenia.

Podobne charakterystyczne szczegółowe informacje o protokołach można otrzymać dla sieci DECnet, OSI, NetBEUI i NetWare. Narzędzia tego typu pozwalają na ocenę zależności czasowych w ruchu pakietów, np. rejestrowanie czasu odbioru danego pakietu z dokładnością zegara PC.

Inne podejście do badania ruchu pakietów reprezentują systemy dla sieci rozległych oparte na protokole SNMP. W systemach tych każdy węzeł rejestruje informacje o pakietach przechodzących przez jego interfejsy sieciowe. Informacje te udostępnia odległym programom analitycznym poprzez sieć, co pozwala na stworzenie obrazu działania całej sieci, lecz nie umożliwia stworzenia charakterystyk czasowych.

Do analizy sieci lokalnej doskonale nadają się urządzenia pierwszego typu. Problemy rozpoczynają się, gdy w ramach jednej sieci mamy kilka segmentów rozdzielonych mostem lub routerem, separującym ruch pomiędzy segmentami i nie przepuszczającym ramek rozgłoszeniowych. Uniemożliwia to np. rejestrację obciążenia serwera Novella, o ile do każdego segmentu nie podłączymy osobnego analizatora. Jednak i w tym wypadku trudno jest stworzyć dokładne charakterystyki czasowe ze względu na brak synchronizacji czasu między poszczególnymi analizatorami.

Nowe technologie sieci lokalnych oparte na przełączaniu spowodowały, że nie istnieje fizycznie takie miejsce w sieci, w którym możemy obserwować cały ruch pakietów segmentu. Fizycznie działanie sieci lokalnej upodobniło się do rozwiązań stosowanych w sieciach rozległych. Analiza takiej sieci w dotychczasowy sposób stała się niewykonalna.

Pomiary zależności czasowych ruchu pakietów w wielosegmentowej sieci lokalnej proponujemy, podobnie jak w SNMP, "rozproszyć" na poszczególne stacje robocze. Każda ze stacji sieci lokalnej prowadzi rejestrację własnej aktywności w sieci (zbiera informacje o wysłanych i odebranych do/z sieci pakietach, z dokładnym określeniem momentów czasów ich nadania i odebrania). Gromadzone informacje są filtrowane, poddawane odpowiedniej, efektywnej kompresji, a następnie w dogodnych momentach przesyłane do "aplikacji" (dedykowanej stacji) prowadzącej monitorowanie sieci. Organizacja procesu gromadzenia informacji o działaniu sieci oparta byłaby na protokole sterowania pomiarami, dla którego proponujemy nazwę Simple LAN Measurement Protocol (Protokół Pomiarów Sieci Lokalnej) przez analogię do SNMP.

2. Protokół pomiarowy sieci lokalnej SLMP

Protokół SLMP umieszczony byłby w architekturze logicznej sieci w warstwie protokołów transportowych; do przesyłu informacji wykorzystywałby podstawowe protokoły transmisyjne danej sieci - np. IPX w Novell NetWare czy IP (lub UDP) w TCP/IP. Wymiana informacji oparta byłaby na mechanizmie dedykowanych dla protokołu punktów dostępu do usług - gniazdek (sockets). Zależnie od zakresu rejestrowanej informacji o pakietach generowałby minimalny lub niewielki ruch pakietów "pomiarowych" w sieci.

2.1. Założenia protokołu pomiarowego

Protokół SLMP byłby protokołem współpracy typu manager-agent pomiarowy. W każdej ze stacji monitorowanej sieci zainstalowany byłby agent, monitorujący aktywność danej stacji i gromadzący o tym informację. Zadaniem managera byłoby skonfigurowanie agentów do przeprowadzenia pomiarów, określenie zakresu rejestrowanej informacji, wyznaczenie momentów rozpoczęcia i zakończenia pomiaru oraz gromadzenie nadsyłanych na bieżąco wyników. Do zadań agenta należałoby gromadzenie informacji o transmisji sieciowej do/z danej stacji sieci, np. przez przejmowanie przerwań od drivera karty sieciowej komputera i odnotowywanie dokładnego czasu wysłania/odebrania każdej ramki, a także obsługa zapytań managera.

Realizacja pomiaru zależności czasowych (monitorowania) sieci odbywałaby się w trzech fazach:

1. Konfiguracja pomiarów obejmowałaby:

- rejestrację konfiguracji działającej sieci (zebranie informacji o wszystkich aktywnych w danej sieci stacjach) i przydzielenie im indywidualnych numerów gniazdek dla obsługi odbioru wyników pomiarów - mogłoby to mieć formę "logowania się" agentów pomiarowych do managera;
- zsynchronizowanie "czasu" obowiązującego w sieci.

2. Prowadzenie pomiarów uruchamiane by było na komendę managera rozsyłaną w sieci ramką typu broadcast; podobnie przeprowadzane by było zakończenie pomiarów. Komenda startu byłaby równocześnie "zaproszeniem" do przesyłania wyników. Każdy z agentów w miarę kompletowania informacji przesyłałby kolejne ramki z wynikami; każda z nich byłaby potwierdzana indywidualnym zaproszeniem do wysłania następnej porcji wyników.

3. Po rozesłaniu ramki zakończenia pomiaru manager ściągałby od agentów informacje końcowe, pozwalające na przetworzenie ramek z wynikami i określenie stanu sieci. Następnie ustawiałby agentów w stan spoczynkowy (nieaktywny).

Pomiary czasu nadania / odebrania pakietów oparte będą na odczycie stanu liczników układu 8254 lub jego odpowiednika na płycie głównej komputera. Układ ten zlicza impulsy o częstotliwości ok. 1.8 MHz, co daje rozdzielczość pomiaru zbliżoną do 0.5 μs. 16-bitowy stan licznika układu uzupełniony byłby 16-bitowym licznikiem programowym, zliczającym przepełnienia licznika układu 8254 (8253). Razem daje to 32-bitowy licznik upływu czasu o okresie przepełnienia ponad 1 godzinę. Stan tego licznika dołączany byłby do informacji o odbiorze / wysłaniu każdej ramki.

Przetwarzanie rejestrowanej informacji

Dla określenia relacji ruchu pakietu konieczne jest zarejestrowanie wybranych danych z nagłówków poszczególnych pakietów. Ponieważ w wybranym horyzoncie czasowym dana stacja wymienia informacje z kilkoma stacjami otoczenia, liczba rejestrowanych adresów sieciowych jest niewielka, a adresy mają stały format. Mogą być one notowane w dedykowanej tablicy (słownik adresów), co pozwala na bardzo efektywną kompresję opartą na indeksie adresu w słowniku.

Jeśli można przyjąć, że rozmiar tablicy adresów ma wielkość do 255 pozycji, to wystarczy indeks o długości 1 bajta dla adresu fizycznego sieci Ethernet lub adresu segment-stacja sieci Novell NetWare. Odpowiada to kompresji 6:1 lub 10:1. Podobny mechanizm wstępnej kompresji może być zastosowany do informacji zawartych w nagłówkach "wewnętrznych" protokołów pakietów; np. informacje z nagłówka ramki Ethernet II lub Ethernet 802.3 mieszczą się na 4 bajtach (indeks adresu odbiorcy - 1 bajt, indeks adresu nadawcy - 1 bajt, długość pakietu - 2 bajty), a ze stanem zegara na ośmiu bajtach. Podobnie informacje adresowe z nagłówka protokołu IPX można zmieścić na 6 bajtach, notując indeksy numerów sieci, adresu stacji i gniazdka z odpowiednich tablic.

W efekcie wszystkie informacje adresowe z nagłówka pakietu wraz z informacją o jego długości i czasie zmieszczą się na 14 bajtach, co pozwala na umieszczenie w jednej ramce Ethernet informacji o 100 odebranych / nadesłanych pakietach. W miarę zwiększania ilości rejestrowanej z nagłówka informacji (rejestracja funkcji pakietów w działaniu sieci) oczywiście rozmiar "opisu" pakietu rośnie, zwiększając liczbę przesyłanych ramek pomiarowych.

Dla "dekompresji" zawartych w opisie rejestrowanych pakietów informacji konieczne jest przesłanie do managera pełnych tablic adresów, gniazdek itp. z każdej stacji (agenta), co można np. zrobić po zakończeniu rejestracji (pomiarów).

2.2. Komendy protokołu SLMP

Z powyższego przeglądu wynika potrzeba zdefiniowania zespołu następujących operacji realizowanych w ramach protokołu pomiarowego:

- **zaproszenie do logowania** - ramka rozgłoszeniowa wysyłana do agentów,
- **zgłoszenie logowania agenta** - ramki wysyłane na "dobrze znany socket" managera z danymi agenta,
- **potwierdzenie logowania** i przygotowanie do pomiarów - ramka odpowiedzi managera na zgłoszenie agenta, wskazująca na indywidualny "adres" gniazdka, rejestracji danych (pakiety z pomiarami mogą być rejestrowane przez kilka różnych komputerów, pełniących funkcję "zastępczych" managerów) oraz określająca zakres (poziom szczegółowości) rejestracji informacji z pakietów stacji:
- **żądanie synchronizacji czasu** agenta z managerem (może być opcjonalne),
- **start pomiarów** - w postaci ramki rozgłoszeniowej,
- **stop pomiarów** - także jako ramka rozgłoszeniowa,
- **wyniki pomiarów** - numerowana sekwencyjnie ramka ze skompresowanymi danymi,
- **potwierdzenie wyników** - zaproszenie do przesłania kolejnej ramki z wynikami,
- **dane pomocnicze** - ramka z tablicami słownikowymi,
- **potwierdzenie danych**,
- **powrót do stanu początkowego.**

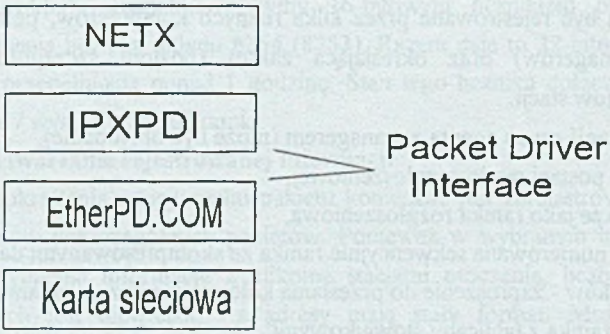
3. Problemy realizacji agenta protokołu pomiarowego

Oprogramowanie agenta powinno mieć formę modułu programowego - demona, instalowanego rezydentnie i przejmującego przerwania sprzętowe lub programowe wybranej warstwy oprogramowania sieciowego stacji w celu śledzenia treści wysyłanych / odbieranych pakietów. Możliwych jest kilka poziomów zainstalowania takiego demona - na poziomie przerwań sprzętowych karty sieciowej, na interfejsie drivera karty sieciowej lub na interfejsie

shella stacji roboczej (czyli na interfejsie programowym modułu IPX lub IP). Po wstępnej analizie jako najbardziej perspektywiczny został wybrany poziom interfejsu drivera karty sieciowej, stąd dalsze prace skupiły się na analizie sposobu działania dwóch popularnych standardów budowy sterowników sieciowych dla systemu operacyjnego MS-DOS: Packet Driver firmy FTP Software Inc. i Open Data-Link Interface firmy Novell Inc., oraz możliwości i sposobie zainstalowania demonów śledzących ruch sieciowy dla tych standardów.

3.1. Standard Packet Driver

Packet Driver - produkt firmy FTP Software, Inc. - dostarcza prostego interfejsu programowego pozwalającego wielu aplikacjom dzielić interfejs sieciowy na poziomie łącza danych (warstwy liniowej). Packet Driver udostępnia funkcje, dzięki którym można m.in. wysłać i odebrać pakiet specyficznego typu, pobrać statystykę dotyczącą packet drivera oraz interfejsu sieciowego. Aplikacja używająca Packet Drivera sama musi uformować ramkę, której postać zależy od klasy sieci. Aplikacja musi zatem dodatkowo oprócz danych zadbać o wstawienie nagłówka, adresu źródła i przeznaczenia do ramki.

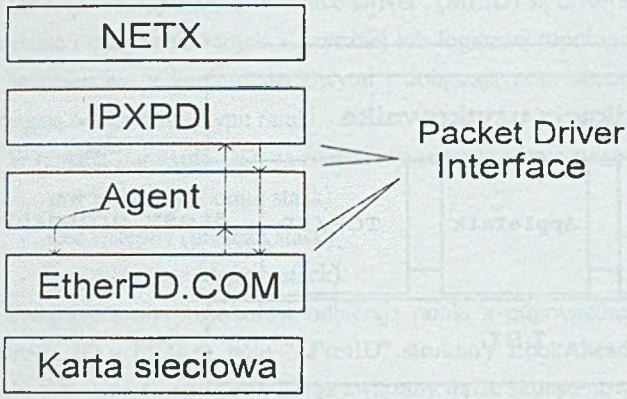


Rys. 1. Struktura oprogramowania stacji roboczej (Packet Driver)
Fig. 1. Workstation software layers (Packet Driver)

Instalacja drivera PD w systemie polega na pozostawieniu w pamięci komputera kodu drivera oraz na zmianie wartości wektora podanego przerwania, tak aby wektor wskazywał na kod obsługi przerwania wewnątrz drivera. Trzy bajty za początkiem procedury obsługi musi się znajdować sygnatura złożona z 8 znaków: "PKT DRVR". Specyfikacja PD przydziela programowe przerwania w zakresie 0x60-0x7F. Aplikacja użytkownika podczas uruchamiania się sprawdza, czy występuje sygnatura pod zadanym numerem przerwania.

Korzystanie z przerwania polega na wywołaniu go, przy czym w rejestrze AH musi być umieszczony kod żądanej operacji. Jedna z funkcji - "access_type" przekazuje driverowi adres

procedury odbierającej. Driver wywołuje procedurę odbierającą, gdy odbierze ramkę z sieci i chce ją przekazać aplikacji użytkownika.

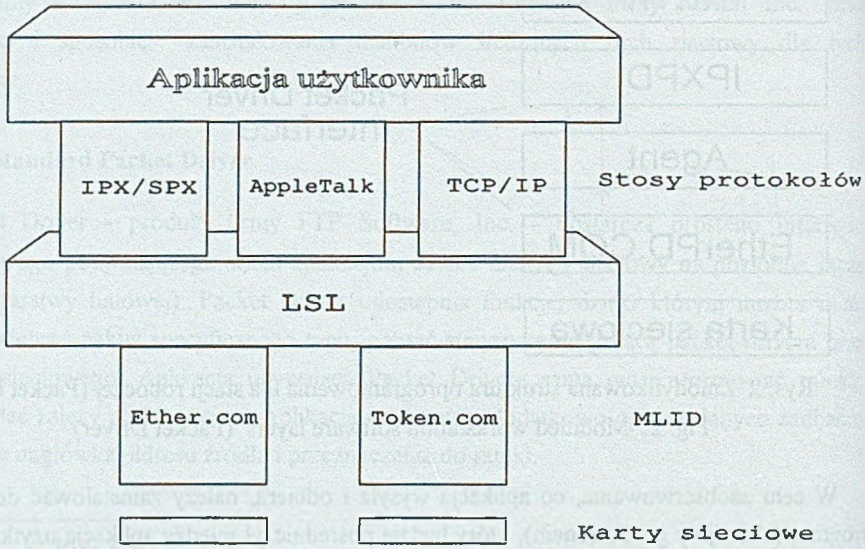


Rys. 2. Zmodyfikowana struktura oprogramowania dla stacji roboczej (Packet Driver)
Fig. 2. Modified workstation software layers (Packet Driver)

W celu zaobserwowania, co aplikacja wysyła i odbiera, należy zainstalować dodatkowy program (nazwijmy go demonem), który będzie pośredniczył między aplikacją użytkownika a driverem PD. Podczas uruchamiania się komputera na początku będzie ładowany driver PD, a następnie zostanie uruchomiony demon. Zadaniem demona jest przejęcie obsługi drivera PD tak, aby uruchamiane w następnej kolejności aplikacje korzystały z funkcji udostępnianych przez demona. Demon powinien posiadać sygnaturę w swoim kodzie, a wszystkie kierowane do niego żądania przełączać do drivera PD, przy okazji notując typ i moment zajścia zdarzenia. Demon musi także podmienić procedurę wykonywaną w przypadku odbioru ramki. Pośrednicząca procedura odbiorcza demona powinna odnotować typ i rodzaj ramki, a następnie powinna przekazać tę ramkę procedurze odbiorczej aplikacji użytkownika. Taka konstrukcja demona pozwala śledzić pracę dowolnej karty sieciowej (niezależnie od mechanizmu komunikacji z kartą sieciową: wspólna pamięć, kanał DMA, porty wejścia/wyjścia) oraz dowolnego protokołu wyższych warstw modelu ISO/OSI, przy założeniu że wszystkie te elementy korzystają z interfejsu Packet Driver.

3.2. Standard Open Data-Link Interface

Standard Open Data-Link Interface (ODI) firmy Novell Inc. jest powszechnie stosowany w programach sieciowych w systemach MS-DOS i MS Windows. W kolejnych punktach zostaną przedstawione czynności podczas wysyłania i odbioru ramek, a następnie zostanie opisany sposób podłączenia demona śledzącego ruch w sieci.



MLID - Multiple Link Interface Driver

LSL - Link Support Layer

Rys. 3. Struktura oprogramowania ODI
Fig. 3. ODI software structure

W ramach tego standardu drivery kart sieciowych są nazywane "Multiple Link Interface Drivers" (MLID). Moduł "Link Support Layer" (LSL) pośredniczy w wymianie informacji między MLID i stosami protokołów (np. TCP/IP, IPX/SPX, LAT i innymi).

Stosy protokołów warstwy sieciowej wysyłają i odbierają dane ponad logiczną lub fizyczną siecią, a także obsługują wybór trasy, usługi połączeniowe oraz zapewniają interfejs dla wyższych warstw modelu ISO/OSI.

Moduł "Link Support Layer" (LSL) obsługuje komunikację między stosami protokołów i driverami kart sieciowych MLID. Ponieważ ODI pozwala fizycznej strukturze sieci na posługiwanie się wieloma różnymi typami protokołów, moduły MLID odbierają ramki przeznaczone dla różnych stosów protokołów, które mogą być obecne w systemie. Przykładowo, jedna sieć Ethernet może obsługiwać jednocześnie następujące protokoły: IPX, TCP/IP, AppleTalk i LAT. Moduł LSL decyduje, który stos protokołu ma otrzymać daną

ramkę. Następnie stos protokołu decyduje, co powinno być zrobione z ramką lub gdzie powinna być ona przesłana. Kiedy stos protokołu wysyła ramkę, wręcza on tę ramkę modułowi LSL, który z kolei przekazuje tę ramkę odpowiedniemu modułowi MLID.

Moduły "Multiple Link Interface Driver" (MLID) są driverami urządzeń, które obsługują wysyłanie i odbieranie ramek z fizycznej lub logicznej topologii. Moduły MLID komunikują się bezpośrednio z kartami sieciowymi i dołączają oraz obcinają nagłówek ramki, a także pomagają w określeniu typu ramki.

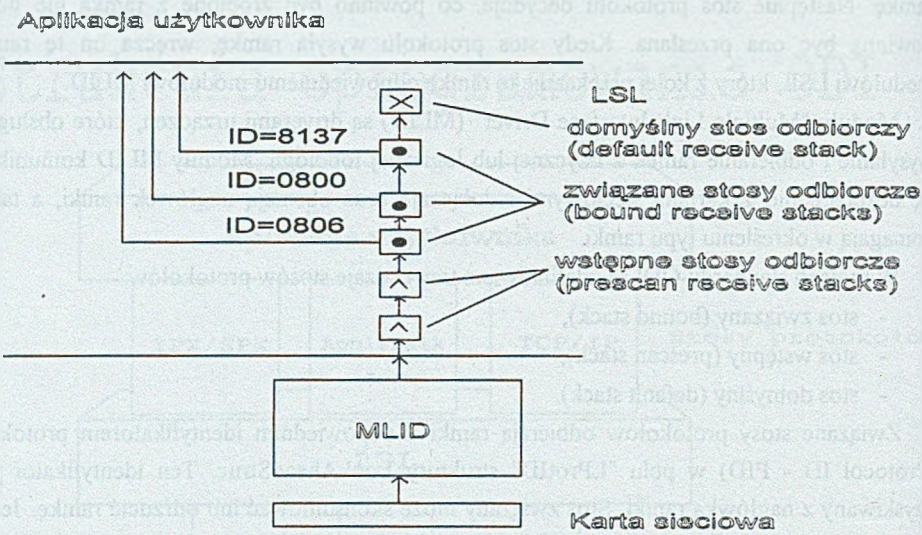
W ramach standardu ODI są zdefiniowane trzy rodzaje stosów protokołów:

- stos związany (bound stack),
- stos wstępny (prescan stack),
- stos domyślny (default stack).

Związane stosy protokołów odbierają ramki z odpowiednim identyfikatorem protokołu (Protocol ID - PID) w polu "LProtID" struktury LookAheadStruc. Ten identyfikator jest uzyskiwany z nagłówka ramki. Stos związany może skonsumować lub odrzucić ramkę. Jeżeli stos protokołu odrzuca ramkę i nie istnieje żaden domyślny stos protokołu dla danej karty sieciowej, to ramka jest niszczona.

Wstępne stosy protokołów przeglądają wszystkie odbierane ramki przez analizę początkowych fragmentów ramki (look ahead method). Wstępny stos protokołu może skonsumować wybrane ramki, a inne przesłać kolejnym wstępnym stosom protokołów lub związanym i domyślnym stosom protokołów.

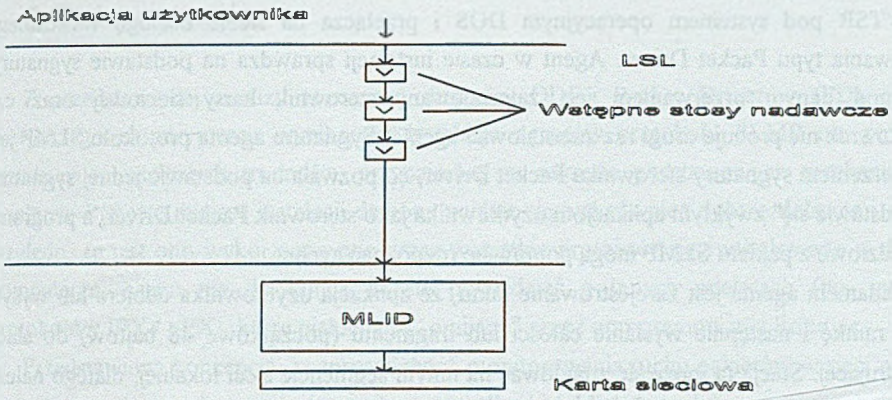
Domyślne stosy protokołów odbierają ramki, które nie zostały skonsumowane ani przez wstępne, ani przez związane stosy protokołów. Jeśli ramka zostanie odrzucona, zostaje ona przekazana kolejnym domyślnym stosom w łańcuchu protokołów. Jeśli żaden stos nie skonsumuje danej ramki, jest ona niszczona i usuwana z systemu.



Rys. 4. Odbieranie ramki
Fig. 4. Frame reception

Prześledźmy przebieg odbioru ramek. Kiedy stos protokołu rejestruje się w obrębie modułu LSL, podaje on modułowi LSL wskaźnik do procedury, która ma zostać wywołana, kiedy moduł MLID odbierze ramkę przeznaczoną dla stosu protokołu. Ta procedura jest nazywana procedurą odbiorczą stosu protokołu. Moduł MLID po odebraniu ramki od karty sieciowej wypełnia strukturę LookAheadStruc i wywołuje funkcję GetStackECB z modułu LSL, aby otrzymać od stosu protokołu bufor, w którym zostaną umieszczone dane ramki. Moduł LSL określa, który ze wstępnych, związanych lub domyślnych stosów otrzyma ramkę i wywołuje dla wybranego stosu procedurę odbiorczą, ze wskaźnikiem do struktury LookAheadStruc, która opisuje odebraną ramkę. Następnie stos protokołu decyduje, czy ramka zostanie odebrana czy odrzucona. W przypadku zaakceptowania ramki stos protokołu tworzy ECB, opisujący zestaw buforów odbiorczych, do których zostanie przesłana treść ramki.

Struktura ECB (Event Control Block) jest strukturą stosowaną do wysyłania i odbierania ramek w standardzie ODI. Zawiera ona wskaźnik do ESR, identyfikator stosu protokołu, ilość danych ramki i inne informacje, potrzebne do prawidłowego działania ODI. Następnie stos protokołu sygnalizuje modułowi LSL, że ten stos skonsumuje ramkę i przekazuje temu modułowi nowo utworzoną strukturę ECB. Moduł LSL przekazuje następnie tę strukturę ECB modułowi MLID. Moduł MLID kopiuje dane ramki do wskazanych buforów, umieszcza strukturę ECB w kolejce i wywołuje procedurę "ServiceEvents". Wtedy moduł LSL wywołuje procedurę "Event Service Routine" (ESR) w obrębie stosu protokołu, co sygnalizuje, że pakiet został odebrany.



Rys. 5. Wysyłanie ramki
Fig. 5. Frame transmission

Aby wysłać ramkę, stos protokołu musi dostarczyć modułowi LSL bufor danych ramki oraz strukturę ECB, która opisuje dane do wysłania. W kolejnym kroku moduł LSL wręcza strukturę ECB każdemu wstępnemu stosowi protokołu (zarejestrowanemu jako stos nadawczy). Ramka może zostać skonsumowana przez taki stos i wtedy jest on zobowiązany do zwrotu przetworzonej struktury ECB. Jeśli ramka nie została skonsumowana lub jeśli w ogóle nie było żadnych wstępnych nadawczych stosów protokołów, moduł LSL uruchamia bezpośrednio procedurę nadawczą w obrębie MLID. Następnie moduł MLID przesyła dane ramki do karty sieciowej, następuje fizyczne wysłanie ramki, a struktura ECB wraz z buforem ramki są zwracane do modułu LSL, niezależnie od tego, czy transmisja zakończyła się pomyślnie, czy wystąpił błąd. W ostatnim kroku moduł LSL wywołuje procedurę ESR, aby poinformować stos protokołu o zakończeniu transmisji.

Istnienie w standardzie ODI wstępnych stosów nadawczych i odbiorczych pozwala na proste zainstalowanie demona w systemie - wystarczy, by zainstalował on dwa wstępne stosy protokołów (nadawczy i odbiorczy), co pozwoli na śledzenie ruchu ramek w obie strony. Te stosy powinny odrzucać wszystkie ramki, żeby nie usunąć żadnej z wymienianych ramek informacyjnych (z wyjątkiem ramek sterujących protokołu SLMP), a przy okazji zanotować typ i czas pojawienia się danej ramki.

4. Wyniki wstępnych eksperymentów

W celu sprawdzenia możliwości instalacji demona na interfejsie protokołów warstwy liniowej napisano dla packet drivera prostego demona, którego zadaniem było kopiowanie wszystkich ramek wysyłanych i odbieranych przez stację roboczą i przesyłanie ich do hipotetycznego komputera rejestrującego działanie danej stacji. Instaluje się on jako program

typu TSR pod systemem operacyjnym DOS i przełącza na siebie obsługę wskazanego przerwania typu Packet Driver. Agent w czasie instalacji sprawdza na podstawie sygnatury, czy pod danym przerwaniem jest zainstalowany sterownik karty sieciowej oraz czy użytkownik nie próbuje drugi raz zainstalować agenta. Sygnatura agenta protokołu SLMP jest rozszerzeniem sygnatury sterownika Packet Driver, co pozwala na podstawie jednej sygnatury "przedstawić się" zwykłym aplikacjom użytkownika jako sterownik Packet Driver, a programy narzędziowe z pakietu SLMP mogą poprawnie rozpoznać agenta.

Zadaniem agenta jest zarejestrowanie faktu, że aplikacja użytkownika odbiera lub wysyła jakąś ramkę i następnie wysłanie całości lub fragmentu (początkowe 48 bajtów) do stacji nadzorującej. Stacja ta może się znajdować na innym segmencie sieci lokalnej, dlatego należy utworzyć pakiet w takim protokole, który jest obsługiwany przez router łączący obie sieci.

Dla celów naszego eksperymentu adres sieciowy odbierającego komputera został „zaszyty" na stałe w programie, do transportu ramek wykorzystano protokół IP. W sytuacji gdy wysyłany jest cały pakiet, należy się liczyć z obcięciem jego końcówki - po dodaniu nagłówka IP sumaryczna długość pakietu może przekroczyć dopuszczalną wielkość ramki w sieci lokalnej.

Standard Packet Driver posługuje się pojęciem uchwytu (ang. handle), który jest używany podczas wysyłania ramki. Agent SLMP podczas wysyłania danych do stacji nadzorującej także musi użyć uchwytu. Istnieje kilka sposobów znalezienia uchwytu: agent może czekać, aż aplikacja użytkownika spróbuje wysłać ramkę, a następnie agent zapamiętuje wartość uchwytu i może jej używać przy retransmisji ramek wysyłanych i odbieranych przez aplikację. Innym sposobem jest sprawdzenie, jakie uchwyty są aktualnie wykorzystywane przez sterownik Packet Driver i wykorzystanie jednego z nich.

W celu kontroli działania demona wszystkie przechodzące przez niego ramki są zliczane w liczniku, którego stan może być wyświetlany na ekranie. Zainstalowanie demona nie zmieniło dla użytkownika działania stacji roboczej, wszystkie programy działały poprawnie, a wyświetlany na komendę *daem -i* licznik ramek pokazywał ich wzrastającą ilość. Podgląd ruchu ramek w sieci programem Etherload uruchomionym na innym komputerze na tym samym segmencie sieci wykazywał istnienie aktywnego nadawcy „Kabel" kierującego ramki na adres 157.158.11.60 (numer IP komputera na innym segmencie sieci), zaszyty w programie.

Działanie eksperymentalnego demona było oczywiście różne od planowanego działania agenta pomiarowego, którego zadaniem jest przetwarzanie monitorowanej informacji i minimalizowanie dodatkowego ruchu ramek w sieci, ale pokazało poprawność sposobu instalacji i może być traktowane jako szkielet do dalszej rozbudowy i testów algorytmów przetwarzania danych agenta.

5. Podsumowanie

Z przedstawionej analizy rozwiązań driverów ODI i Packet Driverów (PD) oraz z powyższego eksperymentu wynika, że instalacja agentów protokołu SLMP nie powinna przedstawiać większych trudności, przynajmniej w zakresie podstawowych mechanizmów działania. Przyjęte miejsce instalacji demona wydaje się być również dobrze dobrane i z tego względu, że jest ono wykorzystywane przez wszystkie protokoły sieci instalowane w danym komputerze, czego nie do końca da się powiedzieć o innych miejscach (np. interfejs protokołów IPX / SPX), które niekiedy są „omijane" przez oprogramowanie firmowe.

Przedstawiona koncepcja „rozproszonego" monitorowania ruchu pakietów w sieci wydaje się możliwa do zrealizowania i powinna umożliwić dokładną analizę czasową obciążenia serwera obsługującego kilka segmentów sieci lokalnej.

LITERATURA

[1] HP NetMetrix Distributed Internetwork Monitoring/Analysis System on Technical Data, Hewlett-Packard Company, 1995.

[2] HP Internet Advisor for LAN - Problem Solving Series, Hewlett-Packard Company,1995.

[3] PC/TCP Version 1.09 Packet Driver Specification FTP Software, Inc.

[4] Novell ODI Specification: NetWare 16-Bit DOS Protocol Stacks and MLIDs, Part Number 107-00078-001, ODI Specification Version 4.00, Document Version V 1.01, 16 May 1995.

[5] EthLoad 1.04 User's Guide, Eric Vyncke.

[6] RFC 1157, Simple Network Management Protocol, D. Case, M. Fedor, M. Schoffstall, C. Davin, 1990.

[7] RFC 1757, Remote Network Monitoring Management Information Base, S. Waldbusser, 1995.

[8] Skrzewski M., Kasprzyk P., Domański A.: Monitorowanie ruchu pakietów w sieci wielosegmentowej. VI Międzynarodowa Wojskowa Konferencja Telekomunikacji i Informatyki, Jabłonna 1997.

Recenzent: Dr inż. Ryszard Winiarczyk

Wpłynęło do Redakcji 8 stycznia 1998 r.

Abstract

Proposition of the distributed network packet flow monitoring system based on SLMP protocol, with the possibility of determining exact packet flow timing relationships has been described. General functions and proposed list of protocol commands have been discussed. The concept of packet flow monitoring, data compression and acquisition, time measurement are presented. Possible places of protocol agent installation in workstation software layers are investigated and Data Link Layer interface was chosen as the most suitable. Technical problems of the protocol agent implementation at Packet Driver and ODI interfaces and some early experiment results are also described.