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

Adam DOMAŃSKI, Piotr KASPRZYK
Politechnika Śląska, Instytut Informatyki

SZEREGOWE PROTOKOŁY WARSTWY LINIOWEJ

Streszczenie. Artykuł opisuje eksperymentalne badanie przepustowości protokołów SLIP i PPP. Protokół SLIP jest nieznacznie szybszy od protokołu PPP. Przy niewłaściwej konfiguracji protokół PPP może dwukrotnie zmniejszyć swą wydajność. Przy protokole PPP może wystąpić różna efektywna szybkość transmisji przy wysyłaniu i odbieraniu danych.

DATA LINK SERIAL PROTOCOLS

Summary. This article describes the experimental throughput estimation of SLIP and PPP protocols. The SLIP protocol is slightly faster than PPP protocol. According to inappropriate configuration the PPP protocol can be up to two times slower. This protocol can also transmit and receive data at different rates through single interface.

1. Wstęp

Transmisja w sieci złożonej pomiędzy jej dwoma węzłami połączonymi fizycznym łączem szeregowym odbywa się z wykorzystaniem protokołów liniowych przeznaczonych do transmisji szeregowej. Protokoły te wykorzystując warstwę fizyczną nie są z nią tak mocno związane jak protokoły stosowane w sieciach lokalnych, gdzie warstwa fizyczna i podwarstwa dostępu stanowią jedność. W przypadku transmisji szeregowych utworzenie warstwy fizycznej w oparciu o kanał cyfrowy, czy kanał analogowy nie wpływa na współpracę protokołów liniowych z warstwą fizyczną.

Do podstawowych protokołów wykorzystywanych w transmisji szeregowej należą **HDLC** (bitowe procedury transmisji zgodne ze standardem ISO, jak również **SDLC** i **ADCCP**), **DDCMP** i **LAPB**. Na bazie protokołu HDLC zbudowane są szeroko rozpowszechnione

w sieciach rozległych protokoły SLIP (Serial Line Internet Protocol) i **CSLIP** (Compressed SLIP) oraz **PPP** (Point to Point Protocol).

Ramka protokołu PPP jest to ramka protokołu HDLC, w której w polu przeznaczonym na adres umieszczone są na stałe same jedynki, w polu sterującym wpisano stałą kombinację bitową "00000011" wskazującą, iż jest to protokół PPP oraz z pola "dane" wydzielono dodatkowe 16-bitowe pole przeznaczone do wskazania rodzaju protokołu wyższych warstw wraz ze wskaźnikiem LCP i NCP. Ramka celem wykrycia błędów transmisji zabezpieczona jest cykliczną nadmiarową sumą kontrolną typu CRC-16.

Protokół SLIP opracowany w ramach projektu ARPANET przeznaczony jest do łączenia po łączu szeregowym systemów sieciowych pracujących z wykorzystaniem protokołu sieciowego IP (Internet Protocol). Budowa jego ramki oparta jest w przybliżeniu na ramce HDLC, z tym że składa się tylko z pól flagi umiejscowionych na początku i na końcu ramki oraz z pola danych. Pole flagi posiada kombinację bitową "11000000" różną od kombinacji flagi ramki HDLC. Ponadto w protokole transmisyjnym nie przewidziano elementów pozwalających na protekcję danych. W polu danych ramki SLIP umieszczane są pakiety IP.

Budowa ramki protokołu CSLIP jest identyczna jak ramki SLIP, z tym że pakiety IP przed umieszczeniem w polu danych są kompresowane, a po rozpakowniu poddawane są dekompresji. Stosowana metoda kompresji, czyli zmniejszenia długości ciągu danych z zachowaniem jego pełnej zawartości informacyjnej, wykorzystuje jedną z odmian algorytmu Huffmana, określoną standardem CCITT V.42 bis - standardem szeroko stosowanym w protokołach modemowych.

Efektywność przedstawionych protokołów zależy od wielu czynników. Między innymi od tego, czy ramki mają stałą, czy zmienną długość, czy są stosowane potwierdzenia wtrącone, czy też nie, czy protokół jest potokowy, czy typu *wyślij i czekaj*, czy stosowane są linie półdupleksowe czy dupleksowe, a także od poziomu i statystycznych charakterystyk błędów transmisji.

Wydajność protokołów liniowych wykorzystywanych do transmisji łączem szeregowym w znacznym stopniu zależy od poziomu usług warstwy fizycznej. Na przykład w przypadku łącza szeregowego zabezpieczonego modemami wykorzystującymi nowoczesne protokoły dające protekcję rzędu 10^9, z adaptacyjnym systemem przystosowującym długość bloku do zmieniającej się w czasie stopy błędów łącza, ze sprawną kompresją, bardziej wydajne będą protokoły najprostsze, tj. takie, które mają najmniejszy narzut bitowy, czyli bez własnej protekcji, które mają najmniejszy narzut czasowy, czyli bez własnej kompresji danych. Natomiast w warunkach nie zabezpieczonego modemami łącza o dużej (10^{-6}) i zmiennej w czasie stopie błędów będzie akurat odwrotnie, te protokoły, które dysponują powyższymi mechanizmami, będą bardziej efektywne. I tak protokół PPP okazuje się mniej wydajny od prostego protokołu SLIP wykorzystywanego w warunkach łącza zabezpieczonego modemami.

Rzeczywiste protokoły i ich implementacje techniczne są często bardzo skomplikowane. Dlatego też, aby można było sprawdzić ich poprawność i wydajność w założonych warunkach, stosuje się, po uprzednim zdefiniowaniu formalnych matematycznych modeli tych protokołów, cały szereg technik symulacyjnych.

1.1. Charakterystyka SLIP

Zaletami protokołu SLIP niechybnie jest jego prostota implementacyjna pozwalająca realizować połączenia TCP/IP. Stosowany jest zwykle do połączeń seryjnych z szybkością 1.2 do 19.2 Kbps.

Protokół SLIP definiuje dwa specjalne znaki: END i ESC. END ma wartość 300 ósemkowo (192 dziesiętnie), a ESC wynosi 333 ósemkowo (219 dziesiętnie). Aby wysłać pakiet, host SLIP uruchamia wysyłanie danych w pakiecie. Jeśli bajt danych jest tym samym kodem co znak END, to wysyłane są dwa bajty sekwencji ESC i 334 ósemkowo (220 dziesiętnie). Natomiast jeśli wysyłanym znakiem jest znak ESC, to wysyłane są dwa bajty sekwencji ESC i 335 ósemkowo (221 dziesiętnie). Po wysłaniu ostatniego bajtu transmitowany jest znak END.

Ponieważ nie ma standardowej specyfikacji protokołu SLIP, nie ma zdefiniowanego maksymalnego rozmiaru pakietu SLIP. Najczęściej jest to 1006 bajtów zawierających IP i nagłówki protokołu transportowego.

Można wskazać na następujące wady protokołu SLIP:

- adresacja - obydwa komputery w połączeniu SLIP muszą znać adres IP. W teraźniejszych systemach liczba adresów IP jest ograniczona. Dlatego jest to jednym z najważniejszych problemów;
- identyfikacja typu - protokół SLIP nie ma pola typu protokołu. Tylko jeden protokół może działać ponad połączeniem SLIP, tak więc w konfiguracji dwóch komputerów DEC działających w TCP/IP i sieci DEC nie ma nadziei na protokół TCP/IP i DECnet;
- korekcja / naprawa błędów - pakiety zostają transmitowane poprzez zwykłe linie telefoniczne, które charakteryzują się dużymi zakłóceniami. Z powodu uzyskiwanej niewielkiej szybkości transmisji rzędu 2400 bps procedura retransmisji jest bardzo kosztowna. Natomiast detekcja błędu nie jest konieczna na poziomie SLIP, ponieważ aplikacje IP powinny wykryć uszkodzone pakiety. Niestety niektóre aplikacje, takie jak NFS, zwykle ignorują sumy kontrolne i zmuszają media sieciowe do wykrywania uszkodzonych pakietów;

- kompresja – w protokole SLIP nie ma kompresji danych, choć kompresja taka przy tak wolnej transmisji jak 2400 bps poprawiłaby szybkość przesyłania danych. Dopiero nowsza wersja protokołu zwana CSLIP umożliwia taką kompresję.

1.2. Charakterystyka PPP

W roku 1994 pojawił się nowy protokół Point-to-Point zwany w skrócie PPP, który jest rozwinięciem wcześniejszego protokołu SLIP. Protokół Point-to-Point jest przeznaczony dla prostych połączeń, które transportują pakiety pomiędzy dwoma końcowymi stacjami roboczymi. Połączenia te realizowane są przez jednoczesne dwukierunkowe operacje i są przyjmowane do dostarczenia pakietów kolejno. PPP łączy różnego rodzaju hosty, bridge i routery. Można wskazać na następujące cechy protokołu PPP:

- hermetyzacja PPP pozwala łączyć ze sobą różnego rodzaju protokoły w ramach jednego połączenia. Takie rozwiązanie pozwala na efektywne wykorzystanie protokołu z różnymi często spotykanymi urządzeniami;
- możliwość automatycznego uzgadniania opcji, zmiany ograniczeń wielkości pakietów, detekcji połączenia pętli zwrotnej i innych zwykłych pomyłek konfiguracyjnych oraz ograniczania połączenia. Inne opcjonalne stosowane ułatwienia to identyfikacja sieci;
- łatwość konfiguracji przez użytkownika lub administratora sieci. Możliwe jest automatyczne połączenie do stacji równorzędnej bez interwencji operatora. Występuje również zewnętrzny mechanizm negocjacji, który reguluje uprawnienia i wymagania kolejnej fazy połączenia;
- wieloprotokołowość, grupowanie informacji w obrębie ramek, które wskazują początek i koniec transmisji;
- autoryzacja danych użytkownika poprzez nazwę i hasło zaraz po ustabilizowaniu się połączenia;
- monitorowanie jakości poprzez narzucenie ograniczenia co do częstości tracenia danych przez protokół;
- negocjacja kompresji pola danych.

2. Teoretyczna analiza wydajności protokołów SLIP i CSLIP

Do transmisji danych protokołu IP przez linie szeregowe (synchroniczne lub asynchroniczne) można wykorzystać protokoły SLIP lub PPP. Poniżej zostanie przedstawiona próba teoretycznego oszacowania wpływu zastosowania protokołu SLIP na wydajność transmisji danych w protokole FTP (opartym o połączenia TCP).

Protokół SLIP wpływa na zmniejszenie wydajności łącza przez zastosowanie dwóch mechanizmów: separacji ramek i przeźroczystości informacyjnej.

Kolejne ramki protokołu IP są oddzielane od siebie przez wprowadzenie między nie bajtu o wartości 192 (SLIP END). Powoduje to wydłużenie o jeden bajt każdej transmitowanej ramki, co zmniejsza przepustowość łącza (przy założeniu że średnia długość ramki wynosi A) do A/(A+1) wielkości początkowej. Załóżmy, że średnia długość ramki to 1500 bajtów, wtedy spadek przepustowości wynosi 1500/1501.

Zapewnienie przeźroczystości protokołu SLIP ze względu na transportowane dane jest osiągane przez zamianę w dwóch z 256 przypadków (dla znaków SLIP ESC i SLIP END) pojedynczego bajtu na sekwencję dwubajtową. Jeśli założymy, że w wejściowym strumieniu danych każda wartość bajtu (od 0 do 255) występuje równie często, to przepustowość kanału maleje o 256/(256+2)=128/129.

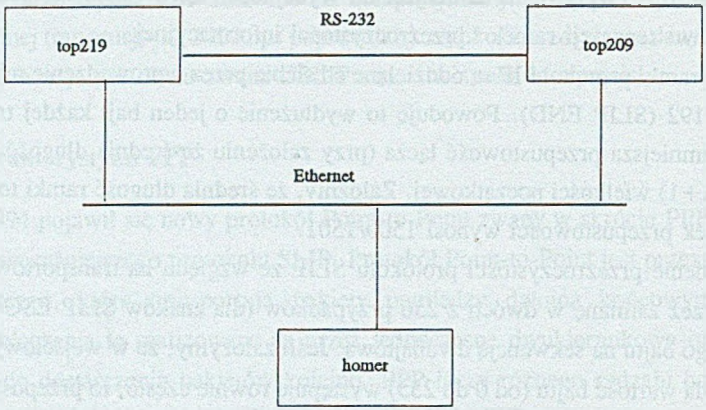
Dane przesyłane za pomocą protokołu TCP opierają się na wykorzystaniu protokołów IP i TCP. Długość nagłówka IP wynosi zwykle 20 bajtów (przy nagłówku bez dodatkowych opcji), a długość nagłówka TCP to kolejne 20 bajtów. Suma ich długości wynosi 40 bajtów, co zmniejsza przepustowość kanału o A/(A+40). Przy zastosowaniu protokołu CSLIP średnia długość nagłówka to 5 bajtów i wtedy przepustowość kanału spada o A/(A+5).

Podsumowując uzyskane wyniki: dla protokołu SLIP możemy się spodziewać spadku przepustowości [1500/1501] * [128/129] * [1500/1540]=0.9658, a dla protokołu CSLIP analogiczne wyliczenia podają wartość [1500/1501] * [128/129] * [1500/1505]=0.9883.

3. Eksperymentalne badanie wydajności protokołów SLIP i PPP

W wyniku przeprowadzonych badań spodziewano się sprawdzić, w jakim stopniu złożoność protokołów SLIP i PPP wpływa na użyteczną przepustowość kanału. Pomiary dokonywano przeprowadzając transmisję plików binarnych o różnej zawartości za pomocą programu FTP. Pozwala to sprawdzić, jaki wpływ mają zastosowane protokoły i ich konfiguracja na rzeczywistą szybkość transmisji przez linie szeregowe. Transmisja plików musiała być tak zorganizowana, aby nie odbywała się jednocześnie w obie strony, gdyż to mogłoby sfałszować wyniki pomiarów.

Stanowisko do testowania wydajności protokołów zostało pokazane na rys. 1. Dwa komputery klasy IBM PC, top219 i top209, pracujące pod kontrolą systemu operacyjnego Linux, są między sobą połączone za pomocą kabla łączącego ich porty szeregowe. Porty te są skonfigurowane na transmisję 8 bitów danych, brak sprawdzania parzystości i jeden bit stopu przy szybkości transmisji 9600 bitów na sekundę. Komputery te są także podłączone do stacjonarnej sieci Ethernet, przez którą komunikują się z resztą sieci Internet, a zwłaszcza



Rys. 1. Topologia analizowanej sieci komputerowej
Fig. 1. The topology of the analysed computer network

z komputerem homer (pracującym pod kontrolą systemu HP-UX), pełniącym rolę koordynatora działań na obu Linuxach. Taka architektura stanowiska pomiarowego została przyjęta ze względu na brak prostej metody skierowania informacji synchronizujących w pewnym protokole przez sieć Ethernet, podczas gdy protokół FTP posługiwałby się między tymi samymi komputerami łączem szeregowym. Zastosowano tutaj algorytm routingu statycznego. Routing oparty na protokole OSPF (Open Shortest Path First) pozwala na definiowanie oddzielnej trasy dla każdego protokołu wyższej warstwy oddzielnie, ale w chwili obecnej nie został on na wykorzystywanych komputerach zainstalowany.

Niska szybkość transmisji portu szeregowego między Linuxami pozwala uniezależnić wartości uzyskiwanych pomiarów od stopnia obciążenia komputerów. W ten sposób można prowadzić pomiary danych bez izolowania komputerów od reszty sieci.

Prace badawcze polegały na przeprowadzeniu serii transmisji za pomocą protokołu FTP przez odpowiednio skonfigurowane łącze szeregowe. W czasie badań zmieniano następujące parametry łącza:

- kompresja nagłówka – włączano lub wyłączano kompresję nagłówka TCP/IP według V. Jacobsona,
- dla protokołu PPP ustawiano różne wartości ACCM (Async-Control-Character-Map), aby sprawdzić, w jakim stopniu wpływa to na szybkość transmisji.

Dla każdego ustawienia dokonywano zestawienia łącza w wybranym protokole i o zadanych parametrach, przeprowadzano serię transmisji plików w obu kierunkach (od komputera top209 do komputera top219 i odwrotnie) i rozłączano połączenie. Mechanizm nadzorujący połączenia, transmisje i rozłączenia został zbudowany tak, aby nie wymagał komunikacji między komputerami top209 i top219, a jego zastosowanie powodowało

zsynchronizowane wykonywanie odpowiednich faz algorytmu na obu Linuxach. Dzięki temu mechanizm ten nie przesyłał informacji synchronizujących między Linuxami, a tylko między każdym Linuxem i komputerem homer. Algorytm działania procesu na Linuxach został przedstawiony w tabeli 1.

Tabela 1

Etapy synchronizacji pracy komputerów

Fazy	top219	top209		homer->top219	homer->top209
A.	połącz	połącz			
B.		sched			sched
C.	sched			sched	
D.	ftp				
E.	sched			sched	
F.		sched			sched
G.		ftp			
H.		sched			sched
I.	sched			sched	
J.	rozłącz	rozłącz			
K.	sched			sched	
L.		sched			sched

Oznaczenie "sched" (ang. schedule - rozkład, plan) oznacza moment synchronizacji między procesami. Na komputerze homer zostaje uruchomiony proces, którego zadaniem jest nawiązywać łączność z oboma Linuxami, zachowując odpowiednią kolejność połączeń. Jest to realizowane przez wywołanie programu "sched" z parametrem, będącym numerem IP tego Linuxa, z którym należy się aktualnie połączyć. Dalszy bieg poleceń na homerze jest wstrzymany do momentu, gdy na wybranym Linuxie nie zostanie uruchomiony program "sched" (tym razem bez parametrów), którego zadaniem jest nawiązanie połączenia TCP z programem "sched" z homera, po którym następuje wymiana komunikatów, rozłączenie połączenia i zakończenie obu programów. Jeśli program "sched" na Linuxie zostanie uruchomiony przed programem "sched" na homerze, czeka on na moment synchronizacji, czyli uruchomienie programu na homerze. To pozwala na zrealizowanie mechanizmu "Rendez-vous", znanego z języka programowania Ada.

W fazie "A" zostaje zestawione połączenie w wybranym protokole (SLIP lub PPP) z odpowiednimi opcjami. Kolejność uruchomienia procesów zestawiających i konfigurujących łącze jest dowolna.

W fazie "B" zachodzi synchronizacja między homerem i komputerem top209 - żadne zadanie nie posunie się do przodu, dopóki oba komputery nie nawiążą ze sobą kontaktu. Po zakończeniu fazy "B" mamy pewność, że na komputerze top209 zostało przygotowane łącze

- uruchomiono demona slattach (dla protokołu SLIP) lub pppd (dla protokołu PPP) oraz ustawiono odpowiednio routing.

W fazie "C" zachodzi synchronizacja między homerem i komputerem top219. Po jej zakończeniu mamy pewność, że obie strony łącza szeregowego są odpowiednio skonfigurowane i przygotowane do pracy oraz że można rozpoczynać transmisję.

W fazie "D" zostaje uruchomiona transmisja plików z komputera top209 do komputera top219, a czas transmisji każdego pliku jest rejestrowany.

W fazie "E" po dokonaniu synchronizacji mamy pewność, że transmisja od komputera top209 do komputera top219 została zakończona, a informacja ta trafia do homera, aby mógł on zezwolić w fazie "F" na rozpoczęcie transmisji przez komputer top209. Ta faza pozwala wykorzystać zestawione łącze tak, aby transmisja FTP przebiegała w każdym momencie tylko w jedną stronę. Gdyby taka transmisja zachodziła w dwie strony, to znaczy komputer top209 przesyłałby plik do komputera top219, a komputer top219 przesyłałby inny plik do komputera top209, wartości zanotowanych czasów transmisji byłyby błędne, zawierałyby łączny czas transmisji w obu kierunkach.

W fazie "F" na pewno komputer top219 zakończył transmisję plików z komputera top209 i przez wykonanie synchronizacji między homerem i komputerem top209 dokonuje się zezwolenia na rozpoczęcie transmisji inicjowanej przez komputer top209.

W fazie "G" komputer top209 transmituje pliki z komputera top219.

W fazie "H" dokonuje się synchronizacja między homerem i komputerem top209, która informuje homera, że top209 zakończył swoje transmisje. Po tej fazie może już nastąpić rozłączenie połączenia przez komputer top209 - łącze nie będzie już używane.

W fazie "I" homer informuje komputer top219, że łącze nie będzie już potrzebne i może przystąpić do jego demontażu (usuwany jest proces demona slattach lub pppd).

W fazach "K" i "L" (kolejność nie jest istotna) komputery top219 i top209 informują homera, że zakończyły demontaż połączenia i można przystąpić do zestawiania następnego połączenia. Algorytm wraca ponownie do fazy "A", ale już z inaczej skonfigurowanym łączem szeregowym.

W czasie każdej transmisji były przesyłane pliki o długości 128KB (131072 bajty) o nazwach:

- "0" - plik z bajtami o wartości 0,
- "255" - plik z bajtami o wartości 255,
- "32" - plik z bajtami o wartości 32 - spacje,
- "16" - plik z bajtami o wartości 16 - do sprawdzenia, jak wpływa parametr ACCM na przepustowość łącza,
- "random" - plik z bajtami o wartościach od 0 do 255, wygenerowanych w sposób pseudolosowy,

- "gzip" - fragment pliku "linux-2.0.32.tar.gz" skompresowanego za pomocą "gzip", co usuwa powtarzające się ciągi bajtów,
- "0-255" - kolejne bajty od 0 do 255, wartości bajtów powtarzają się co 256 bajtów.

Pliki te zostały tak dobrane, by reprezentowały skrajne przypadki: duże uporządkowanie ("0", "255", "32", "16") lub ciągi bardzo nieuporządkowane ("random", "gzip"). Plik "0-255" nie zawiera w sobie powtarzających się sąsiednich bajtów, ale pewne uporządkowanie istnieje, więc jest to przypadek pośredni.

Po przeprowadzeniu eksperymentu uzyskano wyniki przedstawione w tabelach 2 i 3.

Tabela 2

Transmisje do komputera top209 (czas w sek.)

Prot.\Plik	ACCM	"0"	"255"	"32"	"16"	"random"	"gzip"	"0-255"
SLIP+	--------	137	137	137	137	138	138	138
PPP+	00000000	138	138	138	138	140	140	140
PPP+	0000000f	138	138	138	138	140	140	140
PPP+	000000ff	138	138	138	138	140	140	140
PPP+	00000fff	138	138	138	138	140	140	140
PPP+	0000ffff	138	138	138	138	140	140	140
PPP+	000fffff	138	138	138	138	140	140	140
PPP+	00ffffff	138	138	138	138	140	140	140
PPP+	0fffffff	138	138	138	138	140	140	140
PPP+	ffffffff	138	138	138	138	140	140	140
SLIP-	--------	140	140	140	140	141	141	141
PPP-	00000000	141	141	141	141	142	142	142
PPP-	0000000f	281	141	141	141	145	144	145
PPP-	000000ff	281	142	142	142	147	147	147
PPP-	00000fff	282	142	142	142	150	149	149
PPP-	0000ffff	282	142	142	142	152	151	152
PPP-	000fffff	282	142	142	282	154	153	154
PPP-	00ffffff	282	142	142	282	156	156	156
PPP-	0fffffff	282	143	142	282	158	158	158
PPP-	ffffffff	283	142	143	282	160	160	161

W tabeli 2 są podane czasy transmisji (podane w sekundach) plików (o nazwach umieszczonych w górnej części tabeli) z komputera top219 do komputera top209. W tabeli 3 zamieszczone są czasy transmisji z komputera top209 do komputera top219. W pierwszej kolumnie jest podana nazwa protokołu zastosowanego na łączu szeregowym (SLIP lub PPP). Znak "+" po nazwie protokołu sygnalizuje włączenie kompresji nagłówka TCP/IP według algorytmu V. Jacobsona, a znak "-" - wyłącza kompresję. W drugiej kolumnie dla protokołu PPP zmieniana jest wartość Async-Control-Character-Map (ACCM) - pozwala na określenie, które spośród trzydziestu dwu pierwszych znaków ASCII mają być wysyłane przez łącze szeregowe jako sekwencja dwóch bajtów. To pole ma postać

Tabela 3

Transmisje do komputera top219 (czas w sek.)

Prot.\Plik	ACCM	"0"	"255"	"32"	"16"	"random"	"gzip"	"0-255"
SLIP+	--------	137	137	137	137	138	138	138
PPP+	00000000	278	138	138	278	157	156	157
PPP+	0000000f	278	138	138	278	157	156	157
PPP+	000000ff	278	138	138	278	157	156	157
PPP+	00000fff	278	138	138	278	157	156	157
PPP+	0000ffff	278	138	138	278	157	156	157
PPP+	000fffff	278	138	138	278	157	156	157
PPP+	00ffffff	278	138	139	278	157	156	157
PPP+	0fffffff	278	138	138	278	157	156	157
PPP+	ffffffff	278	138	138	278	157	156	157
SLIP-	--------	140	140	140	140	141	141	141
PPP-	00000000	282	142	142	282	161	160	161
PPP-	0000000f	282	143	142	282	160	160	161
PPP-	000000ff	282	142	142	282	160	160	161
PPP-	00000fff	282	142	142	282	161	160	161
PPP-	0000ffff	282	142	142	282	161	160	160
PPP-	000fffff	282	142	142	282	161	160	160
PPP-	00ffffff	282	142	143	282	161	160	160
PPP-	0fffffff	282	142	142	282	161	160	161
PPP-	ffffffff	282	142	142	282	161	160	161

ośmiocyfrowej liczby w układzie szesnastkowym, co pozwala na niezależne maskowanie 32 bitów - każdy z nich określa, czy dany znak ma być podmieniany. Zastosowanie tego mechanizmu pozwala na nie wykorzystywanie kodów DC1 i DC3 do transmisji danych - kody te czasami są stosowane do sterowania przepływem danych i są wtedy określane jako XON/XOFF. W niniejszym eksperymencie dokonano zmian pola ACCM z przyrostem co cztery znaki. Można byłoby zmieniać tę wartość co 1 znak, ale eksperyment trwałby cztery razy dłużej (aktualnie wszystkie transmisje zakończyły się po upływie 13 godzin i 30 minut).

4. Wnioski z uzyskanych wyników

Protokół SLIP użyty na łączu pracującym z szybkością 9600 bitów na sekundę (około 960 bajtów na sekundę - każdy bajt wymaga bitu startu, 8 bitów danych i bitu stopu) według wyliczeń teoretycznych pozwalałby na transmisję plików z szybkością 0.9658 * 960 bajtów na sekundę, co daje przepustowość 927.1 bajtów na sekundę (transmisja pliku 128KB w ciągu 141.35 sekund). Analogiczne wyliczenia dla protokołu CSLIP dają przepustowość 0.9883 * 960 bajtów na sekundę (948.76 bajtów na sekundę) i czas transmisji pliku 128KB

równy 138.15 sekund. Wyniki te są zgodne z eksperymentalnymi pomiarami czasów dla plików "random", "gzip" i "0-255", które spełniały założenia przyjęte podczas teoretycznych wyprowadzeń zależności (warunek występowania każdej wartości bajtu z jednakowym prawdopodobieństwem). Dla plików z jednorodną treścią: "0", "255", "32", "16" to założenie nie było spełnione, dlatego ich transmisja zachodziła szybciej. Po opuszczeniu członu, który jest zależny od częstotliwości występowania wszystkich wartości bajtów w pliku, uzyskujemy poprawne czasy transmisji.

Użycie niezerowej wartości ACCM powoduje zwiększenie czasu transmisji danych w przypadku, gdy w strumieniu danych często pojawiają się bajty, które mają być wysyłane w postaci sekwencji dwubajtowej. Dla transmisji pliku "0" czas ten zwiększa się dwa razy. Po włączeniu kompresji nagłówka TCP/IP według algorytmu V. Jacobsona został także włączony algorytm kompresji danych ramki - przy transmisji pliku "0" z komputera top219 do komputera top209 włączenie kompresji dwukrotnie zmniejszyło czas transmisji.

Dla plików, w których wszystkie wartości bajtów występują z jednakowym prawdopodobieństwem (przy transmisji do komputera top209), widać, że przy nieskompresowanych danych większa liczba bitów "1" w polu ACCM powoduje zwiększenie czasu transmisji pliku.

Protokół SLIP jest niewiele (o 1-2 %) szybszy od protokołu PPP. Konfiguracja protokołu PPP wymaga o wiele większej wiedzy, ale może być wykonywana automatycznie, podczas gdy konfiguracja protokołu SLIP jest prosta i wymaga od administratora niewiele wiedzy, gdyż sam protokół jest prosty. Protokół PPP nadaje się do takich sytuacji, gdzie doświadczony administrator konfiguruje oprogramowanie dla klienta, a klient je tylko uruchamia. Ten protokół jest też bardziej uniwersalny. Uzyskane wyniki nie są symetryczne, co znaczy, że komputery po obu stronach łącza przy pracy w protokole PPP nie były jednakowo skonfigurowane. Do dokładniejszych badań na ten temat należałoby znaleźć w ramach systemu operacyjnego komendę, która pozwoliłaby na pokazanie w kompleksowy sposób ustawień parametrów pracy interfejsu PPP.

Przyczyną niesymetrycznych wyników w tabelach 2 i 3 był fakt, że w protokole PPP istnieją dwa parametry ACCM, jeden dla danych wysyłanych i jeden dla odbieranych. Domyślnie ACCM dla danych wysyłanych ma wartość "ffffffff" (wszystkie znaki ASCII o kodach 0-31 są wysyłane jako sekwencje dwubajtowe), a ACCM dla danych odbieranych wynosi 0 (zostaną zaakceptowane wszystkie znaki). Przy konfiguracji protokołu PPP użytkownik może podać tylko ACCM dla danych odbieranych. W czasie negocjacji parametrów między dwoma komputerami jeden z nich narzucał swoje ustawienia stronie przeciwnej, co powodowało ustalenie niesymetrycznej charakterystyki łącza, mimo wykorzystania jednakowych implementacji protokołu PPP po obu stronach - były wykorzystywane te same pliki binarne, ale działające w różnych dystrybucjach systemu:

Slackware i Red Hat. Z tego powodu różniły się także biblioteki dynamicznie linkowane "libc".

5. Uwagi odnośnie do realizacji eksperymentu

Program "sched" na homerze wykorzystuje funkcję "connect()" w celu podłączenia się do wybranego portu TCP na Linuxie, którego numer IP jest podany jako parametr. Takie rozwiązanie ma pewną wadę - jeśli na Linuxie nie został jeszcze uruchomiony proces "sched", funkcja "connect()" na homerze zwraca błąd, bo nikt nie jest gotowy do odpowiedzi na prośbę o zestawienie połączenia TCP. Dlatego proces "sched" na homerze w przypadku takiej odpowiedzi zamyka swoje gniazdko lokalne, uzyskane za pomocą funkcji "socket()", odczekuje trzy sekundy (blokada przed przeciążeniem komputerów i sieci z powodu wielokrotnych prób nawiązania połączenia) i ponownie próbuje zestawić połączenie TCP. Uruchomienie procesu "sched" na Linuxie powoduje otwarcie dostępu do pewnego portu TCP i rozpoczęcie oczekiwania na próbę podłączenia się do tego portu. W tym przypadku nie występuje kłopot z wielokrotnymi próbami. W bardziej eleganckim rozwiązaniu można byłoby zastosować dodatkowy proces synchronizujący: będzie on odbierał jednocześnie wiele połączeń za pomocą funkcji "select()" lub "poll()", a jeśli zobaczy, że podłączyły się dwa procesy, które chcą się ze sobą skontaktować, uruchomi komunikację między nimi. W takim rozwiązaniu nie jest konieczne wielokrotne testowanie, czy już można się podłączyć do serwera.

W czasie realizacji programu "sched" istnieje konieczność wielokrotnego użycia tego samego numeru portu. Normalnie zamknięte za pomocą funkcji "close()" gniazdko jest nieaktywne przez ok. 1 minutę. Po wprowadzeniu opcji SO_REUSEADDR można było się odwoływać wielokrotnie do tego samego portu bez zwłoki czasowej.

LITERATURA

[1] Romkey J.L.: RFC 1055 - Nonstandard for transmission of IP datagrams over serial lines: SLIP.

[2] Postel J.: RFC 791 - Internet Protocol.

[3] Postel J.: RFC 793 - Transmission Control Protocol.

[4] Jacobson V.: RFC 1144 - Compressing TCP/IP headers for low-speed serial links.

Recenzent: Dr inż. Ryszard Winiarczyk

Wpłynęło do Redakcji 8 stycznia 1998 r.

Abstract

This article describes the results of the experimental throughput estimation of SLIP and PPP protocols (tables 2 and 3). These protocols are located in data link layer and are employed in asynchronous serial lines. The study conditions - link rate at 9600 bps and null-modem cable between two computers running Linux - cause that the way of protocol implementation doesn't bring the influence on the experimental results. The files of miscellaneous contents were transmitted through the link. The SLIP protocol is about one percent faster then PPP protocol and is also easier to configure. There is the possibility to configure PPP in asymmetric way. There is also a need for creating the way to show the current configuration of the PPP interface for Linux.