

Programowanie sieciowe

Instrukcja do laboratorium LAB06

Jeśli używają państwo innych systemów niż maszyny wirtualne udostępnione na serwerze pluton, to należy wykonać Zadanie 0.

UWAGA:

1. W razie problemów z komunikacją w ćwiczeniach (np. błąd "connect: Permission denied") wyczyścić iptables z konta root (komenda 'iptables -F' dla ipv4 i 'ip6tables -F' dla ipv6).
2. Systemy Ubuntu poniżej wersji 18.04 nie wspierają SCTP w narzędziach: netstat i ss

Zadanie 0:

Skonfigurować dodatkowe adresy na interfejsie, który jest używanym do komunikacji:

Dla IPv4: 192.168.100.x/24 i 192.168.101.x/24 gdzie x jest najmniej znaczącym bajtem z istniejącego adresu IPv4

Dla IPv6: fc00:1:1::xy/64, fc00:1:2::xy/64, fc00:1:3::xy/64, gdzie x i y są najmniej znaczącymi znakami z istniejącego adresu IPv6

Sprawdzić czy adresy zostały poprawnie ustawione i jest możliwa komunikacja pomiędzy komputerami z użyciem tych adresów.

Załadować moduł sctp do jądra komendą:

```
sudomodprobe sctp
```

Doinstalować bibliotekę libsctp-dev w systemach wywodzących się z dystrybucji Debian i lksctp-tools-devel w systemach wywodzących się z dystrybucji Redhat. Np. dla Ubuntu:

```
sudo apt-get update  
sudo apt-get install libsctp-dev
```

Np. dla Centos'a:

```
yum install lksctp-tools-devel
```

Ćwiczenie 1: SCTP w trybie one-to-one

Przejść do katalogu CW1

Ćwiczenia wykonujemy w grupach dwuosobowych pomiędzy komputerami lub jeśli to niemożliwe za pomocą interfejsu loopback

1. W terminalu przejść do katalogu CW1 z przykładami:

2. Skompilować i uruchomić programy daytimetcpliv4.c i daytimetcpsrvv4.c (lub, do wyboru, daytimetcpliv6.c, daytimetcpsrvv6.c)
3. Uruchomić komunikację pomiędzy serwerem i klientem. Sprawdzić stan gniazd. Jak należy przerobić przykłady, aby zaobserwować stan ESTABLISHED?
4. Skopiować daytimetcplivX.cdo daytimesctpclivX.c i daytimetcpsrvvX.c do daytimesctpsrvvX.c gdzie X wynosi 4 lub 6 w zależności od dokonanego wyboru w punkcie 1.
5. Przerobić przykłady daytimesctpsrvvX.c i daytimesctpclivX.c, tak aby obsługiwały protokół SCTP w trybie „TCP” - w tym celu w funkcji socket() jako trzeci parametr wpisać IPPROTO_SCTP. Wprowadzić opóźnienie przy zamykaniu asocjacji SCTP. Skompilować przykłady (przy komplikacji dodać bibliotekę sctp: -lsctp), uruchomić komunikację i sprawdzić wymianę pakietów oraz stany gniazd poleceniami tcpdump/wireshark oraz netstat:

```
netstat -pnaS (dla SCTP)
```

```
lsof | grep SCTP
```

Sprawdzić asocjacje SCTP w systemie komendą:

```
cat /proc/net/sctp/assocs
```

Jaka jest różnica pomiędzy komunikacją dla przykładów TCP i SCTP w trybie „one-to-one”?

6. Dla przykładów z rozszerzeniem ‘_04’ – uruchomić komunikację i sprawdzić w ilu segmentach TCP a w ilu segmentach SCTP w warstwie aplikacji zostały przetransportowane dane. Dlaczego jest różnica, pomimo że bufore odbiorcze są takie same. Liczba segmentów wysłanych jest wypisywana przez serwer, natomiast liczba segmentów danych odebranych jest wyświetlana w kliencie.

Proszę sprawdzić, co się stanie, jeśli bufor odbiorczy w procesie klienta (stała MAXLINE) jest mniejszy od bufora nadawczego w serwerze.

Ćwiczenie 2: SCTP w trybie one-to-many

Ćwiczenia wykonujemy w grupach dwuosobowych pomiędzy komputerami lub jeśli to niemożliwe za pomocą interfejsu loopback

Zadanie 2: Protokół SCTP w trybie ‘UDP’ (one-to-many)

1. W terminalu przejść do katalogu CW2 z przykładami. Przykłady implementują komunikację z użyciem protokołu SCTP w trybie „one-to-many”.
2. Skompilować i uruchomić programy sctpclientv4_01.c i sctpservv4_01.c polecienniem make, lub :


```
gcc sctpclientv4_01.c -o sctpc -lsctp
      gcc sctpservv4_01.c -o sctps -lsctp
```
3. Uruchomić serwer i następnie uruchomić dwóch klientów na dwa sposoby w dwóch terminalach:

```
./sctp 192.168.100.x  
./sctp 192.168.100.x streams
```

Dla klienta bez parametru 'streams' tekst do wysłania należy wpisać w formacie:

[n] dowolny tekst

Gdzie 'n' oznacza liczbę całkowitą, która określa numer strumienia, w którym będzie wysłany tekst. W jaki sposób można znaleźć dopuszczalny zakres wartości dla liczby n podglądając komunikację pomiędzy serwerem i klientem programem 'tcpdump' lub 'wireshark'?

Wywołanie z parametrem 'streams' jest przykładem na komunikację wykorzystującą niezależne strumienie dla jednej asocjacji – w pierwszym przykładzie każde 'echo' (gdzie 'echo' to dane wpisane z klawiatury) jest przesłane w jednym strumieniu, w drugim (dla drugiego klienta) to samo 'echo' przesyłane jest na 10 strumieniach jednocześnie i jeszcze jest dublowane.

4. Dla obydwu przypadków sprawdzić stany gniazd po stronie klienta i po stronie serwera programem netstat -S. Sprawdzić asocjacje w systemie komendą:

```
cat /proc/net/sctp/assocs
```

5. Programem lsof sprawdzić ile gniazd SCTP zostało otworzonych:
`lsof | grep SCTP`

i wyniki porównać z punktem 4. Czy są zgodne?

Uwaga: program 'lsof' jeszcze nie obsługuje gniazd SCTP dla opcji '-i'

6. Za pomocą programu tcpdump/wireshark podglądać komunikację pomiędzy serwerem i klientami.

7. Proszę sprawdzić, co się stanie, jeśli bufor odbiorczy w kliencie (stała MAXLINE) jest mniejszy od liczby bajtów w wiadomości wysyłanej do serwera – zmniejszyć stałą MAXLINE, tak by miała wartość mniejszą od długości ciągu znaków, który jest wysyłany do serwera - ponieważ program na przemian zapisuje do i odczytuje z gniazda powinien się zachowywać niepoprawnie, ponieważ nie odczyta w jednej iteracji całego przesłanego komunikatu.

Funkcja stp_recvmsg() ustawia bit MSG_EOR jeśli odbierze cały segment SCTP lub jego ostatnią część. Dzielenie segmentu SCTP jest możliwe jeśli bufor odbiorczy jest zbyt mały, żeby odebrać cały segment SCTP w jednej operacji. Przerobić klienta w taki sposób, aby czytał w pętli dopóki nie odbierze ostatniej części segmentu SCTP (tak, aby klient opróżniał bufor odbiorczy dla każdego odebranego komunikatu SCTP).

Zadanie 3 Program sctpservv4_fork_02.c jest przykładem na wykorzystanie funkcji peel_off() do wydzielenia połączenia (asocjacji) z sesji typu one-to-many do sesji one-to-one.

1. Skompilować i uruchomić programy sctpservv4_fork_02.c i sctpclientv4_01.c :

```
gcc sctpclientv4_01.c -o sctp -lsctp  
gcc sctpservv4_fork_02.c -o sctpsf -lsctp
```

2. Uruchomić serwer i następnie uruchomić dwóch klientów w dowolny sposób w dwóch terminalach:

3. Sprawdzić stany gniazd po stronie klientów i po stronie serwera programem netstat
4. Programem lsof sprawdzić ile gniazd SCTP zostało otworzonych po stronie serwera:
`lsof | grep SCTP`
5. Za pomocą programu tcpdump/wireshark podglądać komunikację.
6. Podmienić funkcje czytające w funkcji str_echo w programie sctpserv4_fork_02.c na sctp_recvmsg i sprawdzić, jakie bity w fladze są ustawiane.

Ćwiczenie 3: Powiadomienia o stanach protokołu SCTP

Ćwiczenia wykonujemy w grupach dwuosobowych pomiędzy komputerami lub jeśli to niemożliwe za pomocą interfejsu loopback.

Zadanie 4: Odczytywanie stanów protokołu SCTP w trybie ‘UDP’ z poziomu aplikacji

1. W terminalu przejść do katalogu CW3 z przykładami. Przykłady są rozbudowaną wersją przykładów z CW2 (one-to-many) o dodatkowe opcje powiadamiania o zdarzeniach dla asocjacji SCTP.
2. Skompilować i uruchomić programy sctpclientv6_02.c i sctpservv6_06.c .
3. W oddzielnym terminalu po stronie serwera uruchomić sesję programu tcpdump (lub wireshark):

```
tcpdump -i eth0 -nv port 7
```

(jeśli komunikacja odbywa się w obrębie komputera zamienić interfejs na lo)

4. Uruchomić serwer i następnie uruchomić na drugim komputerze dwóch klientów w dwóch terminalach (w dowolnych trybach) i użyć dla każdego klienta różnych adresów IPv6:

```
./sctpclientv6_02 fc00:1:1::X - np. na pierwszym terminalu  
./sctpclientv6_02 fc00:1:2::X - np. na drugim terminalu
```

5. Zaobserwować informacje o zmianie stanów po stronie serwera i wyświetlenie informacji o adresach asocjacji po stronie klientów.

6. Po stronie serwera usunąć adres fc00:1:1::X :

```
ipaddr del fc00:1:1::X/64 dev eth0
```

zaobserwować zachowanie – sprawdzić, czy wciąż komunikacja pomiędzy klientami i serwerem jest możliwa i po jakich adresach

7. Sprawdzić stany gniazd po stronie klienta i po stronie serwera. Po jakich adresach ma miejsce komunikacja?

8. Po stronie serwera dodać adres fc00:1:1::X :

```
ipaddr add fc00:1:1::X/64 dev eth0
```

i usunąć adres po stronie klienta:

```
ipaddr del fc00:1:1::X/64 dev eth0
```

zaobserwować zachowanie – sprawdzić, czy wciąż komunikacja pomiędzy klientami i serwerem jest możliwa i po jakich adresach.

9. Sprawdzić stany gniazd po stronie klienta i po stronie serwera. Po jakich adresach ma miejsce komunikacja?

10. Dodać adres po stronie klienta:

```
ipaddradd fc00:1:1::X/64 dev eth0
```

Zmienić parametry protokołu SCTP w jądrze systemu klienta i serwera (komendy wykonać z poziomu root'a):

```
echo 1 > /proc/sys/net/sctp/addip_enable  
echo 1 > /proc/sys/net/sctp/addip_noauth_enable
```

i powtórzyć punkty od 5 do 8.

Pytania sprawdzające:

1. Ile minimum gniazd dla protokołu SCTP jest wymaganych do komunikacji z X klientami dla
 - a. Trybu one-to-one
 - b. Trybu one-to-many
 - c. Trybu ‘tcp’
 - d. Trybu ‘udp’
2. Opisz procedurę nawiązania połączenia dla protokołu SCTP
3. Opisz procedurę kończenia połączenia dla protokołu SCTP
4. Jaka jest struktura pakietu SCTP?
5. Jakie są najważniejsze pola nagłówka pakietu SCTP?
6. Stany protokołu SCTP
7. Różnice pomiędzy SCTP i TCP dla trybu one-to-one
8. Różnice pomiędzy SCTP i UDP dla trybu one-to-many
9. W jaki sposób można utworzyć nową asocjację w protokole SCTP?
10. Ile maksymalnie strumieni może być przesyłanych w asocjacji SCTP? Jaką maksymalną liczbę strumieni może obsługiwać asocjacja SCTP biorąc pod uwagę strukturę nagłówka SCTP?
11. Czym różni się użycie funkcji connect() od sctp_connect() dla protokołu SCTP
12. Czym różni się użycie funkcji bind() od sctp_bind() dla protokołu SCTP
13. Jak rozpoznać w protokole SCTP, że zostały odebrane wszystkie fragmenty pakietu.
14. Jakie podstawowe parametry opisują asocjację SCTP?