Programowanie sieciowe

Instrukcja do laboratorium LAB04

Polecenia wykonujemy w systemie ubuntu.

Protokół UDP

Zadanie 1.

1. Przeanalizować kod programów daytimeudpcliv6.c i daytimeudpsrvv6.c, które implementują usługę DAYTIME dla protokołu UDP:

2. Skompilować przykłady serwera i klienta usługi DAYTIME dla protokołu UDP: daytimeudpsrvv6.c i daytimeudpcliv6.c .

3. W dwóch terminalach uruchomić komunikację pomiędzy serwerem i klientem używając interfejsu loopback lub w grupach pomiędzy różnymi komputerami. W oddzielnym terminalu uruchomić program netstat/ss (ss -6uap/netstat --inet6 -upa) i sprawdzić stany gniazd dla serwera i klienta. Za pomocą programu tcpdump/wireshark podglądnąć wymianę pakietów pomiędzy serwerem i klientem. (tcpdump -i lo port 13)

4. W programie klienta daytimeudpcliv6.c, w funkcji main() podmienić wywołanie funkcji dt_cli() na wywołanie funkcji dt_cli_connect(), skompilować program do innej nazwy i powtórzyć punkt 3. Jakie różnice pojawiają się w działaniu programu z funkcja dt_cli() w porównaniu z działaniem programu z funkcją dt_cli_connect()? Jakie informacje o stanach gniazd wyświetlane są dla różnych typów klientów?

5. Zaobserwuj różnice w działaniu programów klientów z punktu 3 i 4 (w zależności od użycia funkcji dt_cli() lub dt_cli_connect()), gdy łączą się z maszyną, na której nie działa serwer. Za pomocą programu tcpdump/wireshark podglądnąć wymianę pakietów pomiędzy serwerem i klientem. (np. tcpdump -i lo -v port 13 or icmp6). Dlaczego wyświetlają różne komunikaty o błędach? Które komunikaty pochodzą od ustawionego czasu oczekiwania na gnieździe opcją SO_RECVTIMEO, a które od komunikatów ICMP. Która funkcja zwraca błąd "Connection refused" dla funkcji dt_cli_connect()? Która funkcja zwracała błąd " Connection refused" dla protokołu TCP?

6. Powtórzyć punkty 3 i 4 dla serwera daytimeudpsrvv6_lossy.c, który nie odpowiada na wszystkie zapytania - symuluje zagubienie pakietu. Łączenie z serwerem powtórzyć kilka razy (przynajmniej trzy) dla każdego z przypadków. Zauważyć, że proces klienta z funkcją dt_cli_connect() jeśli zostanie zawieszony, to jedynym sposobem na przerwanie procesu klienta jest wysłanie sygnału, który zamknie proces (np. przez sekwencję klawiszy CTRL-C) - funkcja dt_cli_connect() nie ma zabezpieczenia przed zagubieniem pakietu i w nieskończoność oczekuje na odpowiedź od serwera. Skopiować mechanizm zabezpieczenia przed zagubieniem pakietu z funkcji dt_cli() (ustawienie opcji SO_RECVTIMEO i ponowienie czytania z gniazda) do funkcji dt_cli_connect(). Sprawdzić poprawność wykonanych zmian.

Zastosowanie jakiej funkcji pozwala na odbieranie błędów asynchronicznych ICMP przez gniazdo UDP, np. informację, że w węźle docelowym serwer nie działa?

Zadanie 2. Implementacja czasu oczekiwania na odpowiedź od serwera za pomocą sygnału SIGALRM. W programie daytimeudpcliv6.c po odkomentowaniu linii nr 20 (definicja stałej M_ALARM) do implementacji czasu oczekiwania na gnieździe używany jest sygnał SIGALRM. Przeanalizować kod i porównać działanie programu z zakomentowaną i odkomentowaną linią nr 20 (**UWAGA**: działanie programów powinno się różnić, gdy występuje zagubienie pakietu).

Zmienić program w ten sposób, aby miał tę samą funkcjonalność, ale bez użycia pętli for do wysyłania i odbierania pakietów:

A. Zastosować flagę SA_RESTART w funkcji sigaction(), tak aby funkcja recvfrom() nie była przerywana po odebraniu sygnału SIGALRM.

B. Gniazdo zadeklarować jako zmienną globalną i wysyłać komunikaty funkcją sendto() z funkcji obsługi sygnału.

Zadanie 3. Przeanalizować kod programu daytimeudpcliv_options.c. Skompilować program i uruchomić komunikację łącząc się z serwerem z punktu 1. Jest to przykład, który implementuje mechanizm pobierania adresu docelowego z nagłówka protokołu IP (IPv4 lub IPv6) za pomocą funkcji recvmsg(). Przerobić program klienta i serwera w ten sposób, by serwer mógł ustawiać pole IP_TTL (i/lub IPV6_UNICASTHOPS) za pomocą funkcji sendmsg()(*trudne*) lub opcji gniazd (*łatwe*), a klient mógł dodatkowo odbierać tę opcję z nagłówka protokołu IPv4 (i/lub IPv6).

Do przygotowania na następne zajęcia (LAB05):

1. Wiadomości z laboratoriów #3 i #4.

2. Wiadomości z wykładów od 3 do 5.

Pytania sprawdzające:

1. Jaka jest domyślna reakcja procesu na otrzymanie sygnałów (gdy w programie nie są przechwytywane sygnały) SIGCHILD, SIGPIPE, SIGURG
2. DO czego służą flagi SOCK_NONBLOCK i SOCK_CLOEXEC w funkcjach socket() i accept()
3. Do czego służy funkcja connect() dla protokołu UDP?
4. Kiedy proces może odbierać błędy asynchroniczne ICMP na gnieździe UDP?
5. Do czego służy funkcja listen() dla protokołu UDP?
6. Jakie funkcje sieciowe należy wywołać po stronie serwera, aby można było odebrać pakiet od klienta:

a) dla protokołu TCP?

b) dla protokołu UDP?

7. Jakie funkcje należy wywołać po stronie serwera, aby można było wysłać pakiet do klienta:

a) dla protokołu TCP?

b) dla protokołu UDP?

8. Jakie funkcje sieciowe należy wywołać po stronie klienta, aby można było odebrać pakiet od serwera:

a) dla protokołu TCP?

b) dla protokołu UDP?

9. Jakie funkcje sieciowe należy wywołać po stronie klienta, aby można było wysłać pakiet do serwera:

a) dla protokołu TCP?

b) dla protokołu UDP?

10. W jaki sposób w serwerze UDP(gniazdo niepołączone) i TCP można uzyskać informacje:

a) o adresie źródłowym pakietu?

b) o porcie źródłowym pakietu?

c) o adresie docelowym pakietu?

d) o porcie docelowym pakietu?

11. W jaki sposób w serwerze UDP(gniazdo niepołączone) i TCP można uzyskać informacje:

a) o adresie źródłowym pakietu?

b) o porcie źródłowym pakietu?

c) o adresie docelowym pakietu?

d) o porcie docelowym pakietu?

12. W jaki sposób w kliencie UDP(gniazdo niepołączone) i TCP można uzyskać powyższe informacje?

13. Jakie kroki należy wykonać, aby odebrać informację z nagłówka pakietu IP funkcją recvmsg()?

14. Jak powinny zostać ustawione adresy w funkcjach wysyłających dla protokołu UDP i TCP?

15. Jak działa serwer współbieżny dla protokołu UDP.