Programowanie sieciowe

Instrukcja do laboratorium - LAB 10

Biblioteka libpcap

Zadanie 1. Program libpcap1.c demonstruje użycie wybranych funkcji z biblioteki libpcap:

- sprawdzić, czy jest zainstalowany pakiet libpcap-dev (komendy dotyczą systemu Ubuntu, w systemie Centos należy użyć komendy yum i sprawdzać pakiet libpcap-devel):
`dpkg –l libpcap-dev (yum list libpcap-devel)`
jeśli pakiet libpcap-dev nie jest zainstalowany to należy go zainstalować:
```
sudo apt-get update
sudo apt-get install libpcap-dev (yum install libpcap-devel)
```

- przeanalizować i skompilować program (dla gcc dołączyć bibliotekę libpcap – opcja '-l pcap' lub do kompilacji użyć programu make) i uruchomić program:

```
gcc libpcap.c  -o pcap  -lpcap

./pcap
```

1) co program realizuje? 2) jakie opóźnienie jest ustawione, 3) jaka jest maksymalna długość pakietów przekazywanych z jądra do programu, 4) gdzie znajduje się informacja o oryginalnej długości pakietu, 5) jaki filtr ustawiony jest domyślnie, 6) jak przekazać do programu inny filtr, 7) jakie opcje przyjmuje program, 8) jak włącza/wyłącza się tryb 'promiscous' karty sieciowej (flaga PROMISC w komendach ip i ipconfig)

2) zmienić program dodając następujące funkcjonalności (1 pkt.):
a) dodać wyświetlanie adresów IPv6,
b) dodać wyświetlanie portów i flag dla protokołu TCP

3) **Nieobowiązkowe:** dodać statystyki przechwytywanego ruchu (w zależności od czasu): liczba pakietów, liczba bajtów (generowany ruch w bit/s), dla IPv4, dla IPv6, dla TCP, dla UDP. (0.5 pkt.)

4) podmienić funkcję przechwytującą pakiety na pcap_next_ex() uwzględniając zwracane błędy przez tą funkcję. (0.5 pkt)

5) **Nieobowiązkowe:** podmienić funkcję do przechwytywania pakietów pcap_next() na pcap_loop(). (0.5 pkt)

Gniazda surowe IPv4 i IPv6

Zadanie 2. Omawiany na wykładzie program ping, na który składają się następujące pliki źródłowe: main.c, ping.h, init_v6.c, proc_v6.c, proc_v4.c, send_v4.c, send_v6.c, sig_alrm.c, readloop.c oraz plik Makefile, jest uproszczona implementacją programu ping dla protokołu IPv4 i IPv6 (patrz wykład 10):

1) przejść do katalogu ping, przeanalizować kod, skompilować program komendą make

2) zmienić właściciela i prawa dostępu do programu ping w taki sposób, aby można było go uruchomić z użytkownika student bez używania komendy sudo (Podpowiedź: ustawić prawa i właściciela analogicznie do systemowego ping'a)

3) przetestować program dla różnych typów adresów (IPv4 i IPv6) i dostępnych opcji

4) zmienić kod programu, dodając opcję -i, pozwalającą zmienić częstość wysyłania pakietów ICMP_ECHO_REQUEST (0.5 pkt)

5) zmienić kod w ten sposób, aby można było wysyłać pakiety ICMP na adresy rozgłoszeniowe (broadcast) jeśli podamy opcję –b (dla tej opcji należy ustawić na gnieździe opcję SO_BROADCAST) (0.5 pkt)

6) zmienić kod w ten sposób, aby przy wysyłaniu na adresy rozgłaszania grupowego (multicast) można było wybrać interfejs, na którym są wysyłane komunikaty ICMP za pomocą opcji -I – dodać odpowiednie opcje dla IPv4 i IPv6. Przetestować działanie programu. (0.5 pkt)

Zadanie 3. Znaleźć informację jak działają funkcje bind() i connect() dla gniazd surowych IPv4 i IPv6 oraz jakie struktury adresowe są używane dla tych funkcji w przypadku gniazd surowych.

Pytania sprawdzające:

1. Jakie główne kroki należy wykonać w programie korzystającym z biblioteki libpcap do podglądnięcia zawartości pakietu?

2. Jaki parametr interfejsu należy znać, aby otworzyć interfejs za pomocą API biblioteki libpcap?

3. Czym różni się funkcja pcap_open_live() od funkcji pcap_create()?

4. Czym różni się funkcja pcap_open_live() od funkcji pcap_open_offline()?

5. Czym różni się funkcja pcap_open_live() od funkcji pcap_activate()?

6. Wymień dwa parametry, które można ustawić na uchwycie do przechwytywania pakietów pcap_t w bibliotece libpcap()

8. Jak interpretować następujący filtr biblioteki libpcap:

    ip6 tcp and port 80 and tcp[13:1] & 0x3 != 0

9. Czym różną się funkcje **pcap_dispatch**() i **pcap_loop**()?

10. Jakie informacje przekazywane są w strukturze **struct pcap_pkthdr**

11. Czym różnią się funkcje **pcap_dispatch() i pcap_next()**

12. Czym różni się gniazdo typu SOCK_RAW od gniazda SOCK_DGARM i SOCK _STREAM?

14. Jakie funkcje gniazdowe można użyć do gniazd typu SOCK_RAW?

15. Jeśli ustawiono opcję IP_HDRINCL, to jakie pola nie muszą być wypełniane w nagłówku IP:

    a) w systemach typu UNIX

    b)w systemie LINUX

16. Jakie są różnice przy wysyłaniu pakietów typu SOCK_RAW dla protokołów IPv4 i IPv6.

17. Jakie są różnice przy odbieraniu pakietów typu SOCK_RAW dla protokołów IPv4 i IPv6.

18. Jakiego typu pakiety są zawsze przekazywane do gniazda surowego w systemie typu UNIX

19. Jakiego typu pakiety są zawsze przekazywane do gniazda surowego w systemie LINUX

20. Jak można filtrować pakiety, które są odbierane na gnieździe surowym, z poziomu programu bez użycia API do iptables?