Tugas Kecil 1 IF3130 Jaringan Komputer

FLOW CONTROL

Tifani Warnita (13513055)
Asanilta Fahda (13513079)

K-01

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2015

Mengapa dalam tugas kecil ini digunakan UDP, bukan TCP?

Karena TCP sudah memiliki *flow control* sedangkan UDP tidak. Karena itu, kami menggunakan UDP untuk belajar bagaimana cara kerja *flow control* dengan mengimplementasikannya sendiri secara manual.

Jelaskan perbedaan TCP dan UDP!

	TCP	UDP
Koneksi	Berbasis koneksi (dibangun dengan *three-way handshake*)	Tidak berbasis koneksi
Kecepatan transfer	Lebih lambat	Lebih cepat
Keterandalan	Pesan dijamin sampai dengan lengkap dan sesuai urutan	Pesan tidak dijamin ketersampaiannya
Penanganan kesalahan	Dilakukan pengecekan dan pemulihan dari kesalahan	Dilakukan pengecekan namun tidak terdapat metode pemulihan dari kesalahan
Flow control	Ada	Tidak ada
Weight	*Heavyweight*	*Lightweight*
Penggunaan oleh protokol lain	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, TFTP, SNMP, RIP, VOIP
Metode transfer	Data dikirim sebagai *byte stream* yang tidak memiliki batasan terdefinisi	Data dikirim sebagai paket individu yang memiliki batasan terdefinisi
Penggunaan umum	Email, file transfer	Game, video streaming

Mengapa minimum *upper limit*/batas atas minimum harus lebih kecil dari jumlah karakter yang bisa ditampung dalam *buffer*?

Minimum *upper limit* harus lebih kecil dari jumlah karakter yang bisa ditampung di dalam *buffer* karena saat mengirim XOFF kepada *transmitter* untuk memberi tahu bahwa minimum *upper limit* telah tercapai, bisa terjadi kondisi di mana *transmitter* sudah telanjur mengirim karakter atau XOFF yang dikirim mengalami hambatan karena pada dasarnya pengiriman sinyal XOFF ke *transmiter* juga membutuhkan kurun waktu tertentu. Apabila saat mengirim karakter *buffer* sudah penuh, karakter yang dikirim dapat menimpa *buffer* yang sudah terisi sehingga dapat mengacaukan isi *buffer*. Maka dari itu, sebaiknya minimum *upper limit* harus lebih kecil dari jumlah karakter yang bisa ditampung di dalam *buffer*.

Petunjuk Kompilasi Program

1. Buka folder IF3130_T1_K1-G16.
2. Buka terminal pada direktori tersebut.
3. Ketik `make`
4. Hasil kompilasi (*file executable*) dapat dilihat di folder bin (receiver dan transmitter).

Petunjuk Penggunaan Program

1. Buka folder IF3130_T1_K1-G16/bin.
2. Buka terminal pada direktori tersebut.
3. Buka file executable receiver menggunakan terminal dengan parameter port. Contoh: `./receiver 4040`
4. Buka file executable transmitter menggunakan terminal dengan parameter IP address receiver, port, dan nama file. Contoh: `./transmitter 192.168.56.129 4040 kucing.txt`

Sumber Referensi

https://www.cs.rutgers.edu/~pxk/417/notes/sockets/udp.html
http://www.binarytides.com/c-program-to-get-ip-address-from-interface-name-on-linux/

Kode yang digunakan:

transmitter.c

```
1.  struct sockaddr_in remaddr;
2.  int sockfd, slen=sizeof(remaddr);
3.  int recvlen;
4.
5.  sockfd = socket(AF_INET, SOCK_DGRAM, 0)
6.
7.  memset((char *) &remaddr, 0, sizeof(remaddr));
8.     remaddr.sin_family = AF_INET;
9.     remaddr.sin_port = htons(service_port);
10.    if (inet_aton(server, &remaddr.sin_addr)==0) {
11.        fprintf(stderr, "inet_aton() failed\n");
12.        exit(1);
```

```
13. }
14.
15. if(sendto(sockfd, &ch, 1, 0, (struct sockaddr *)&remaddr, slen)==-1) {
16.    perror("sendto");
17.    exit(1);
18.  }
19. recvlen = recvfrom(sockfd, &ch, 1, 0, (struct sockaddr *)&remaddr, &slen);
```

receiver.c

```
1.  struct sockaddr_in myaddr;     // Our address
2.  struct sockaddr_in remaddr;    // Remote address
3.  socklen_t addrlen = sizeof(remaddr); // Length of addresses
4.  if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
5.              perror("Cannot create socket\n");
6.              return 0;
7.          }
8.  memset((char *)&myaddr, 0, sizeof(myaddr));
9.           myaddr.sin_family = AF_INET; // Address family used when set up the
    socket
10.         myaddr.sin_addr.s_addr = htonl(INADDR_ANY); // Address for socket
11.         myaddr.sin_port = htons(atoi(argv[1])); // Port number of socket
12.
13.         if (bind(sockfd, (struct sockaddr *)&myaddr, sizeof(myaddr)) < 0) {
14.             perror("bind failed");
15.             return 0;
16.         }
17. ifr.ifr_addr.sa_family = AF_INET; //Type of address to retrieve
18.      strncpy(ifr.ifr_name,"eth0", IFNAMSIZ-1); //Copy the interface name in the
    ifreq structure
19.     ioctl(sockfd, SIOCGIFADDR, &ifr); //Get the IP address
20.           printf("Binding  pada  %s:%s\n",  inet_ntoa((  (struct  sockaddr_in
    *)&ifr.ifr_addr )->sin_addr), argv[1]);
21. recvlen = recvfrom(sockfd, &ch, 1, 0, (struct sockaddr *)&remaddr, &addrlen);
```

Pembagian Kerja dalam Kelompok

	Tifani Warnita	Asanilta Fahda
Transmitter	20%	80%
Receiver	80%	20%
Laporan	50%	50%