Tugas Kecil 1 - IF3130 Jaringan Komputer

Flow Control

Dipersiapkan oleh

Kelompok Haha

13514029 - Muhammad Farhan Majid
13514047 - Bervianto Leo P
13514095 - Muhammad Az-zahid Adhitya Silparensi

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2016

DAFTAR ISI

Pembahasan Pertanyaan **3**
Mengapa menggunakan UDP, bukan TCP? 3
Perbedaan TCP dan UDP 3
Mengapa Minimum Upperlimit lebih kecil dari jumlah karakter yang bisa ditampung dalam Buffer? 4
Pentunjuk Kompilasi Program **5**
Petunjuk Penggunaan Program **6**
Pembagian Kerja dalam Kelompok **7**
Referensi **7**

1. Pembahasan Pertanyaan

1.1. Mengapa menggunakan UDP, bukan TCP?

Dikarenakan TCP sudah memiliki *Flow Control* sedangkan UDP belum memilikinya dan dalam tugas ini mengeksplorasi mengenasi *Flow Control* yang akan ditambahkan pada *layer* aplikasi yang menggunakan UDP.

1.2. Perbedaan TCP dan UDP

Perbedaan TCP dan UDP dapat dilihat melalui tabel berikut :

No	Kategori	TCP	UDP
1.	Koneksi	Protokol berorientasi koneksi	Protokol tidak berorientasi koneksi (*connectionless*)
2.	Penggunaan	Sesuai untuk aplikasi yang membutuhkan tahan uji (*realibility)* tinggi dan waktu transmisi yang tidak menjadi suatu hal yang kritis. Contoh: Web, SSH, FTP, telnet, SMTP, IMAP/POP[3]	Sesuai untuk aplikasi yang membutuhkan kecepatan, keefektifan transmisi, seperti *game*. Sifat *stateless* UDP berguna juga untuk server yang membutuhkan menjawab kueri-kueri kecil dari banyak klien. Contoh: tunneling/VPN, media streaming, games
3.	Digunakan oleh Protokol-Protokol, seperti :	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, TFTP, SNMP, RIP, VOIP
4.	Pengurutan paket data	TCP mengurutkan paket data dalam urutan yang telah dispesifikan.	UDP tidak memiliki keterurutan seperti setiap paket merupakan independen. Jika membutuhkan pengurutan, dalam layer aplikasi perlu mengaturnya.
5.	Kecepatan	Lebih lambat dibandingkan UDP.	UDP lebih cepat dikarenakan pemulihan galat tidak dilakukan. UDP merupakan protokol “best effort”.
6.	Tahan uji (Realibility)	Ada jaminan bahwa data yang ditransfer tetap utuh dan tiba dengan urutan yang sama saat dikirim.	Tidak ada jaminan bahwa pesan atau paket yang dikirim akan tercapai sama persis.

7.	Ukuran Header	20 byte	8 byte
8.	Data Flow Control	TCP melakukan Flow Control. TCP membutuhkan tiga paket untuk membuat koneksi socket, sebelum data dapat dikirim.	UDP tidak memiliki pilihan untuk Flow Control.
9.	Pengecekan Error	TCP melakukan pengecekan galat dan memperbaiki galat tersebut. Paket yang galat ditransmisi kembali oleh sumber ke tujuan.	UDP melakukan pengecekan galat tetapi mengabaikan paket yang galat tersebut.
10.	Acknowledgement	Terdapat segmen Acknowledgement	Tidak memiliki Acknowledgement
11.	Handshake	SYN, SYN-ACK, ACK	Tanpa handshake (Protokol *connectionless*)
12.	Isi	1. Sequence Number 2. ACK Number 3. Data Offset 4. Reserved 5. Control Bit 6. Window 7. Urgent Pointer 8. Options 9. Padding 10. Check Sum 11. Source Port 12. Destination Port	1. Length 2. Source Port 3. Destination Port 4. Check Sum

1.3. Mengapa Minimum Upperlimit lebih kecil dari jumlah karakter yang bisa ditampung dalam Buffer?

Agar tidak terjadi *overflow* pada *buffer*. Dengan menetapkan *minimum upper-limit* yang lebih kecil dibandingkan kapasitas *buffer*, maka *receiver* akan meminta *transmitter* untuk menghentikan pengiriman data sebelum *overflow* terjadi. Jika *minimum upper-limit* lebih besar dibandingkan kapasitas *buffer*, maka *overflow* sudah terjadi dan data yang dikirimkan dapat hilang.

2. Pentunjuk Kompilasi Program

Melakukan kompilasi dapat menggunakan *Makefile* untuk mempermudah. Adapun berikut ini detail melakukan kompilasi :

a. Kompilasi *Transmitter*

   i. Menggunakan *Makefile*

      Dalam menggunakan *Makefile* sangat mudah, lakukan perintah berikut untuk kompilasi :

      ```
      $ make transmitter
      ```

      - Catatan : Pastikan Anda pada posisi *root* dari program atau pada posisi direktori yang memiliki *Makefile*. Jangan pindahkan *Makefile*, karena akan memberikan kegagalan kompilasi. Program hasil kompilasi akan ada pada direktori *bin*.

   ii. Manual

      Secara manual dapat Anda lakukan dengan memasukan perintah ini :

      ```
      $ g++ transmitter.cpp -o transmitter
      ```

      - Catatan : Lakukan perintah ini pada posisi yang terdapat *transmitter*. Hasil program akan ada pada direktori yang sama dengan kode sumber.

b. Kompilasi *Receiver*

   i. Menggunakan *Makefile*

      Dalam menggunakan *Makefile* sangat mudah, lakukan perintah berikut untuk kompilasi :

      ```
      $ make receiver
      ```

      - Catatan : Pastikan Anda pada posisi *root* dari program atau pada posisi direktori yang memiliki *Makefile*. Jangan pindahkan *Makefile*, karena akan memberikan kegagalan kompilasi. Program hasil kompilasi akan ada pada direktori *bin*.

   ii. Manual

      Secara manual dapat Anda lakukan dengan memasukan perintah ini :

      ```
      $ g++ receiver.cpp -o receiver
      ```

      - Catatan : Lakukan perintah ini pada posisi yang terdapat *receiver*. Hasil program akan ada pada direktori yang sama dengan kode sumber.

c. Kompilasi Semua (*Transmitter* dan *Receiver*)

   i. Menggunakan *Makefile*

      Dalam menggunakan *Makefile* sangat mudah, lakukan perintah berikut untuk kompilasi :

```
$ make all
```

- Catatan : Pastikan Anda pada posisi *root* dari program atau pada posisi direktori yang memiliki *Makefile*. Jangan pindahkan *Makefile*, karena akan memberikan kegagalan kompilasi. Program hasil kompilasi akan ada pada direktori *bin*.

ii. Manual

Secara manual dapat Anda lakukan dengan memasukan perintah ini :

```
$ g++ receiver.cpp -o receiver
$ g++ transmitter.cpp -o transmitter
```

- Catatan : Lakukan perintah ini pada posisi yang terdapat *transmitter* dan *receiver*. Hasil program akan ada pada direktori yang sama dengan kode sumber.

3. Petunjuk Penggunaan Program

Anda dapat menggunakan program ini dengan mudah, adapun berikut untuk menjalankan program dan parameter yang harus diberikan :

a. Menjalankan *Receiver*

Pada sistem operasi Linux Anda dapat melakukan perintah berikut ini :

```
$ ./receiver [PORT]
```

Contoh :

```
$ ./receiver 20000
```

Catatan :

- *PORT* yang dimasukan ini merupakan *port* pada komputer penerima yang akan bersedia menerima koneksi.

b. Menjalankan *Transmitter*

Pada sistem operasi Linux Anda dapat melakukan perintah berikut ini :

```
$ ./transmitter [IP Tujuan atau Hostname] [PORT] [Lokasi Berkas]
```

Contoh :

```
$ ./transmitter 192.168.0.7 20000 test.txt
```

Catatan :

- *IP Tujuan* yang dimasukan ini merupakan alamat IP atau hostname yang akan dikirim sebuah berkas.
- *PORT* yang dimasukan ini merupakan *port* pada komputer tujuan yang akan bersedia menerima koneksi.
- *Lokasi Berkas* yang dimasukan ini merupakan alamat berkas yang ingin dikirim dan nama berkas yang akan dikirim.

c. Prosedur Penggunaan

Program yang dibuat saling berkaitan, oleh karena itu agar program berjalan sesuai dengan yang diharapkan adapun prosedur yang harus dilakukan sebagai berikut :

1. Jalankan *Receiver* (Lihat cara menjalankan *Receiver*)
2. Jalankan *Transmitter* (Lihat cara menjalankan *Transmtter*)
   Receiver ini harus dipersiapkan untuk menerima koneksi agar *Transmitter* dapat mengirim informasi berkas pada komputer tujuan tersebut yang sudah membuka koneksi.

4. Pembagian Kerja dalam Kelompok

Adapun berikut ini pembagian kerja dalam pembuatan program ini :

- 13514029 - Muhammad Farhan Majid :
  Membuat *Receiver* dan mengisi laporan.
- 13514047 - Bervianto Leo P
  Membuat *Receiver* dan mengisi laporan.
- 13514095 - Muhammad Az-zahid Adhitya Silparensi
  Membuat *Transmitter* dan mengisi laporan.

Referensi

1. https://github.com/masphei/flow-control-udp/blob/master/Control.c
   Bagian kode yang digunakan yaitu pada pengisian prosedur *rcvchar* dan *q_get* serta cara mengimplementasi thread pada Receiver dengan sedikit perubahan.
2. http://www.diffen.com/difference/TCP_vs_UDP - Perbedaan TCP dan UDP
3. http://stackoverflow.com/questions/5330277/what-are-examples-of-tcp-and-udp-in-real-life diakses pada Senin, 7 November 2016