

# F74046022\_陳冠仁\_電腦通訊網路\_lab2

### 環境參數

Ubuntu 16.04 (WSL)

Java 1.8.0\_162

### 使用方式

(程式在 Multicast/bin/和 Multithread/bin/)

(程式碼在 Multicast/src/和 Multithread/src/)

java MulticastServer ip port /path/to/file

java MulticastClient ip port

以編號缺失計算封包遺失率

java MultithreadServer ip port /path/to/file

java MultithreadClient ip port /path/to/dir

以封包重複計算封包遺失率 (可靠傳輸)

先開啟 n 個 client 再開啟 server 選定要傳輸之檔案

### 心得

這次作業我有嘗試以 Multicast 實作可靠傳輸，過程中我發現這比起一對一的 UDP 可靠傳輸要複雜得多。由於 Multicast 無法選擇發送對象，因此 client 端會收到很多無效封包，但收到無效封包後直接重送 request 又容易導致效能低落，所以 client 必須要有一個決定忽略或是重送的機制。

### ### 實驗結果

#### Multicast Server

```
Bash on Ubuntu on Windows
bin$ java MulticastServer 225.0.0.1 8888 ../../test/a.pdf
bin$ _
```

[0] 0: bash\* "KJCHEN" 20:56 07-May-18

#### MulticastClients

```
連接 Ubuntu
bin$ java MulticastClient 225.0.0.1 8888
Loss rate : 0.23489933 %
bin$

bin$ java MulticastClient 225.0.0.1 8888
Loss rate : 0.23825504 %
bin$

bin$ java MulticastClient 225.0.0.1 8888
Loss rate : 0.23825504 %
bin$
```

[0] 0: bash\* "BACKUP" 20:58 07-May-18

## MultithreadServer

```
Bash on Ubuntu on Windows
bin$ java MultithreadServer 192.168.43.131 8080 ../../test/a.pdf
Port 61907 assigned
Port 61909 assigned
Port 61908 assigned
Port 61907 Transfer complete
Port 61909 Transfer complete
Port 61908 Transfer complete
Server closed
bin$
```

[0] 0: bash+ "KJCHEN" 21:20 07-May-18

## MultithreadClients

```
Ubuntu
bin$ java MultithreadClient 192.168.43.131 8080 ../1/
File transfer complete
1210928 bytes transfered
Loss rate : 0.0 %
bin$

bin$ java MultithreadClient 192.168.43.131 8080 ../2/
File transfer complete
1210928 bytes transfered
Loss rate : 0.0 %
bin$

bin$ java MultithreadClient 192.168.43.131 8080 ../3/
File transfer complete
1210928 bytes transfered
Loss rate : 0.0 %
bin$
```

[0] 0: bash+ "BACKUP" 21:20 07-May-18