1. 實驗日期：

2020/04/30

1. 實驗名稱
2. ITS Exp. 9: 路徑MTU之發現
3. ITS Exp. 10: Network Disturbance網路干擾分析
4. ITS Exp. 11: Error Control錯誤控制
5. 問題與答案

**Exp. 9: 路徑MTU之發現**

1. **說明為什麼Figure 9.9的封包長度為196？**

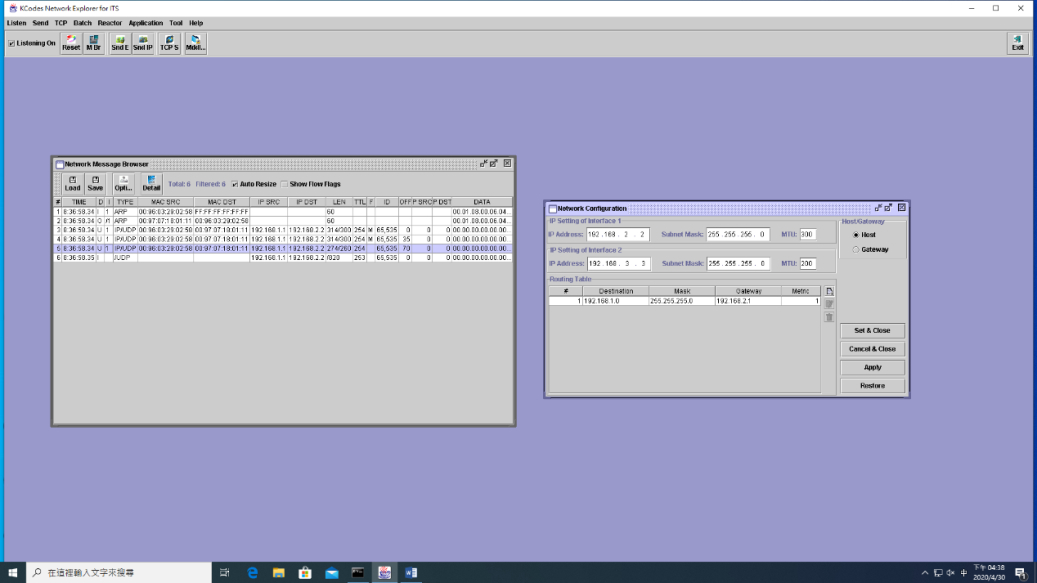
IP封包內容:

一個frame的表頭內容會包含，12 bytes控制碼/4 bytes 來源IP/4 bytes 目的IP 加起來總共20 bytes。

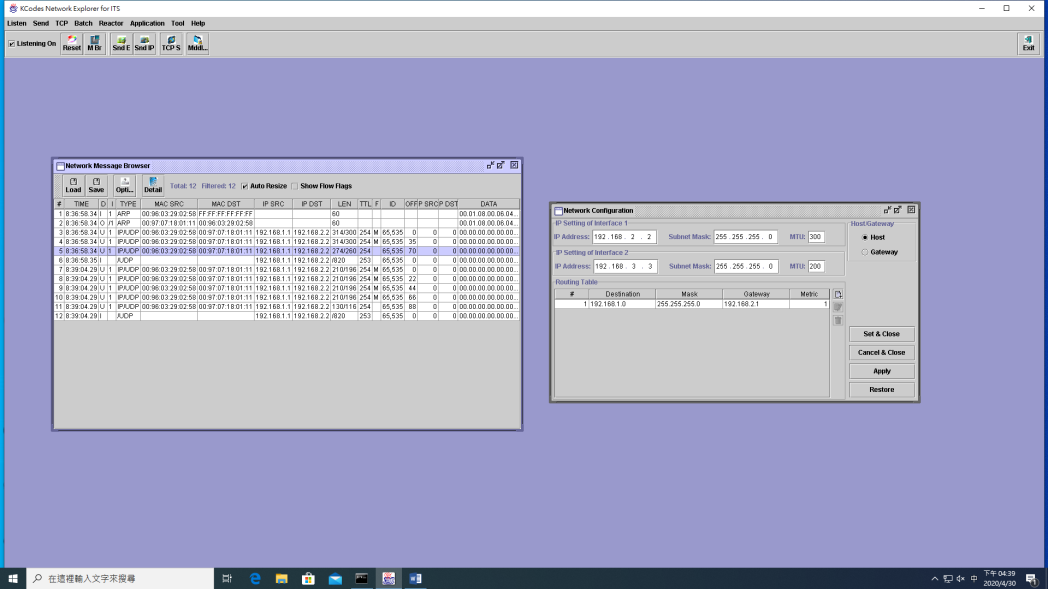
當封包被切割時，每個封包前表頭長度為20 bytes，而最大能整除的資料數為176，176+20得196。

1. **在網路拓樸A裡，當你發送MTU=200的封包和MTU=300的封包時，它在傳送時，最大的差異在哪？**

* **MTU=300**



* **MTU=200**

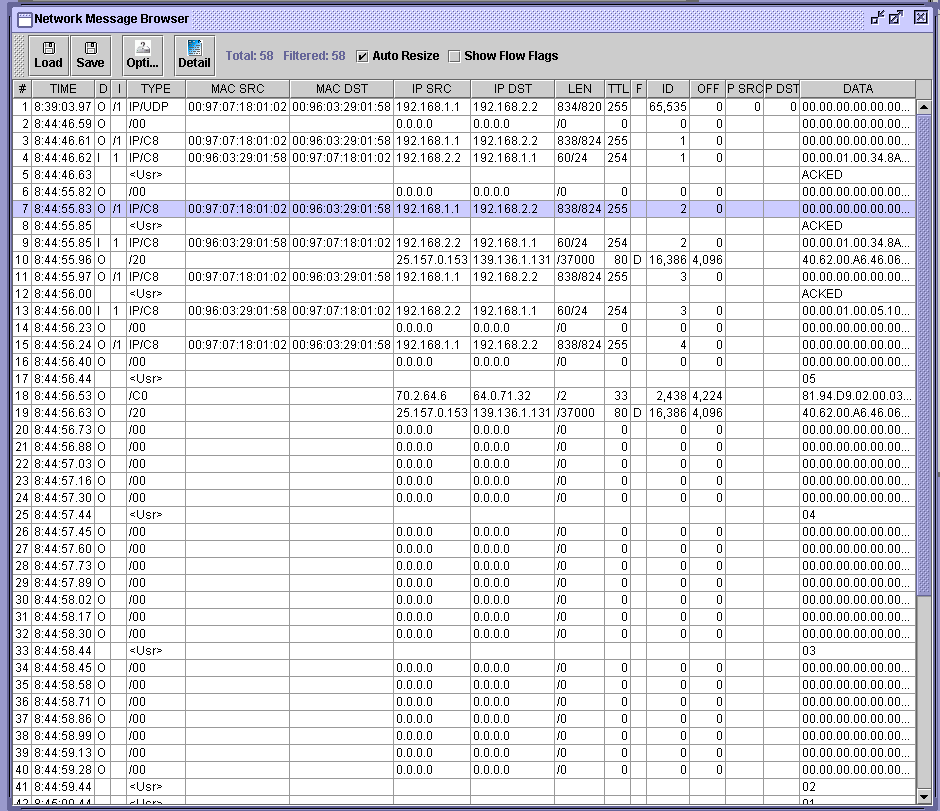


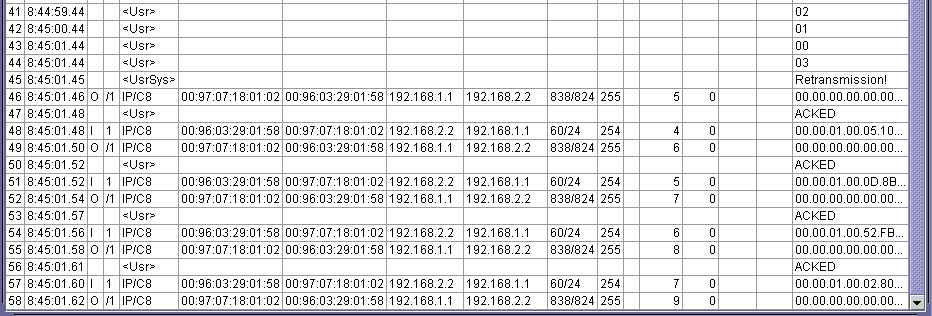
當MTU為300時，因為300扣掉header 20 bytes 等於280，280能被8整除，

因此傳送的LEN為280；但當MTU為200時，扣掉表頭 20 bytes 為 180，180無法被8整除，只有176才行，176+20(加回扣掉的表頭)=196，因此傳送的LEN為196。MTU為最大傳輸單元，MTU=300傳輸的封包一次可以傳比較多，但容易比網路傳輸路徑的最小MTU還大，之後便開始執行對大塊封包做切割的動作，進而減少系統的壓力，封包在傳輸的時候會不斷的重新組合，反而降低傳輸效能。

**Exp. 11: Error Control錯誤控制**

1. **說明各驗收項目Message Brower中所顯示訊息的意義。**
2. **Idle RQ without Packet Identification**

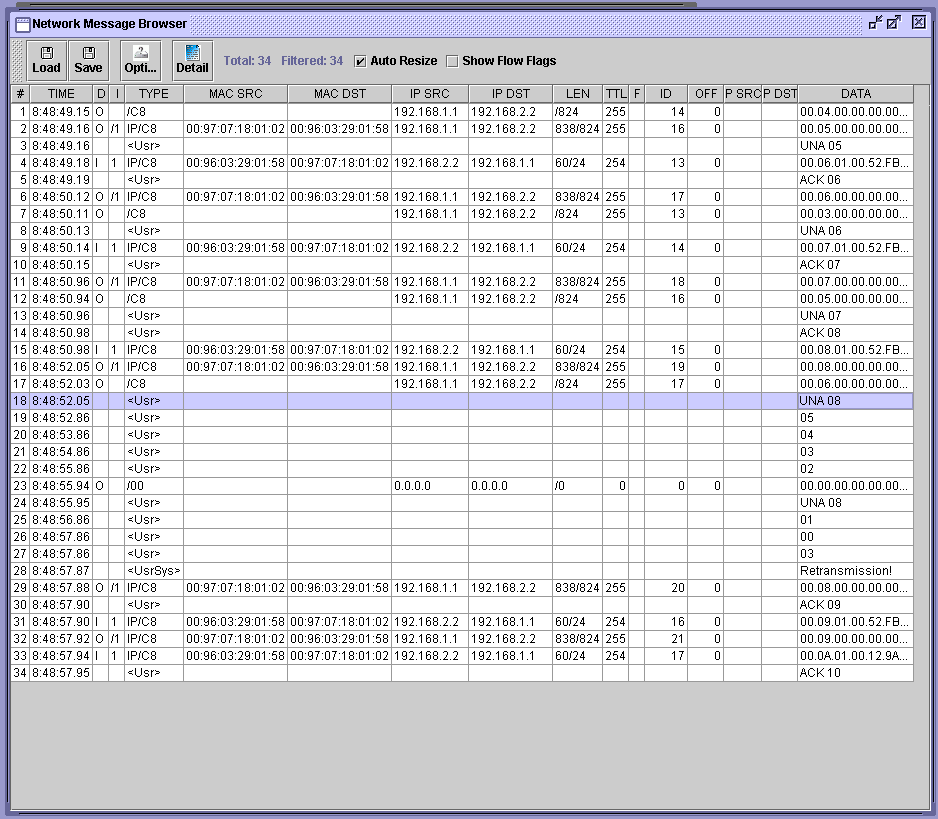




由圖可以發現第一次傳輸沒有延遲，第二次傳輸延遲4秒，第三次延遲7秒，

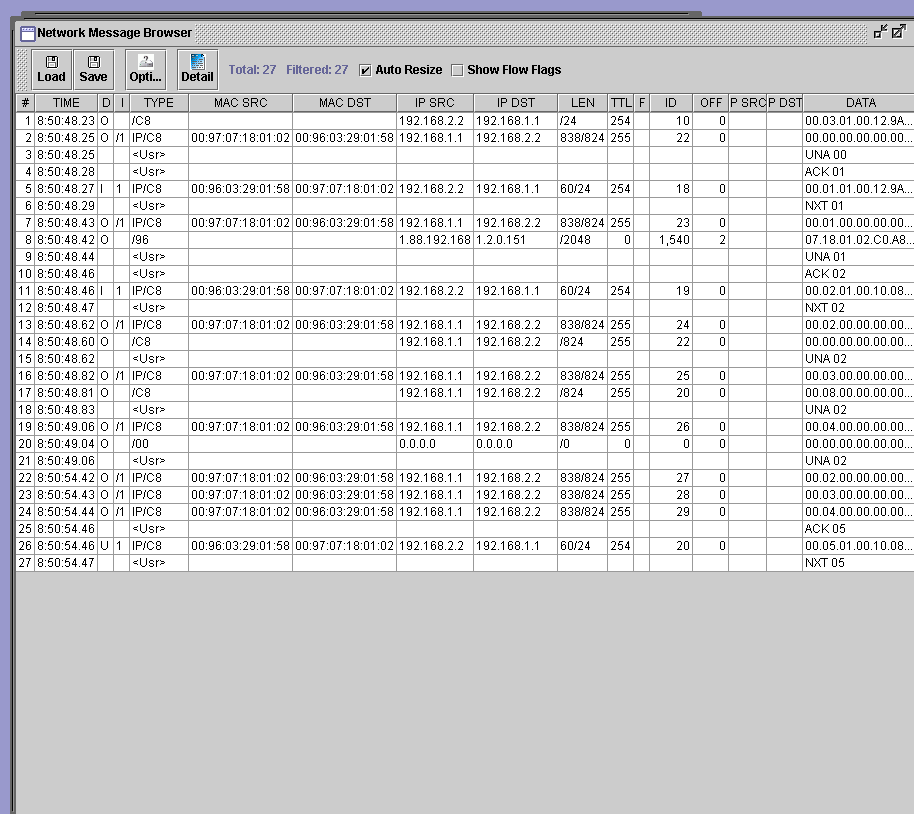
第四次移失在第五次ACK。

1. **Idle RQ**



ACK用於確認資料有無正確的傳輸到接收端，當收到ACK時(在封包有效時)才回傳下一個封包。與Idle RQ without Packet Identification 的差別為知道掉哪一個封包，由上圖可知掉的是第四個封包，所以在03的時候做Restransmission。

1. **Continuous RQ**



Continuous RQ 不需要等transmission 結束再重傳，發現封包遺失(不管中間有沒有收到ACK)沒有便會立即重傳。

1. **往返時間(RTT, Round Trip Time)是指一個封包從發送端出發，到某個網路上的節點或目的端，並收到回應所需要的平均時間。我們通常將重送時間(retransmission time)設為兩倍RTT。如果我們將重送時間拉長或變短，對網路傳輸會有什麼樣的影響？試著改寫發送端的程式並討論其影響。**

重送時間如果拉長，這樣網路的等待時間將會跟著變長；重送時間如果變短，可能會發生已經快到的封包被迫重新傳送，造成浪費網路資源且使得網路阻塞。

1. **比較一下，Continuous RQ是否比Idle RQ更有效率？在Idle RQ的方式中，最多可以扣留幾個IP datagram？試著在packet lost的情況發生時，多送幾個IP datagram看看。**

Continuous RQ 不需要等transmission 結束再重傳，發現封包遺失便會立即重傳，Idle RQ的使用率極低，而且其等待時間比Continuous RQ還要長，所以Continuous RQ比Idle RQ還更有效率。Idle RQ最多扣留4個IP datagram。

1. **結果討論**

更加清楚封包互相傳送的形式為何，Message Brower中所表示的訊息也更加了解。

1. **補充資料**

<https://zh.wikipedia.org/zh-tw/ACKv>

<https://notfalse.net/23/mtu>

<https://blog.csdn.net/mgxcool/article/details/73028534?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.nonecase&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.nonecase>