WiresharkLab\_IP

建議:把比例放到150%圖才會清楚

組別:A10

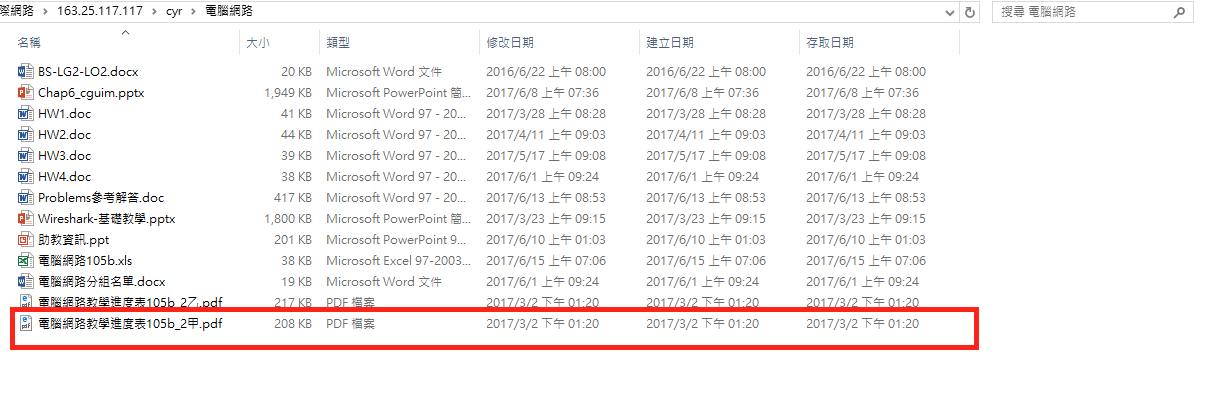
組長:資管二甲 B0444132 張仁樵

組員:資管二甲 B0444101 李積舜

組員:資管二甲 B0444138 潘楚華

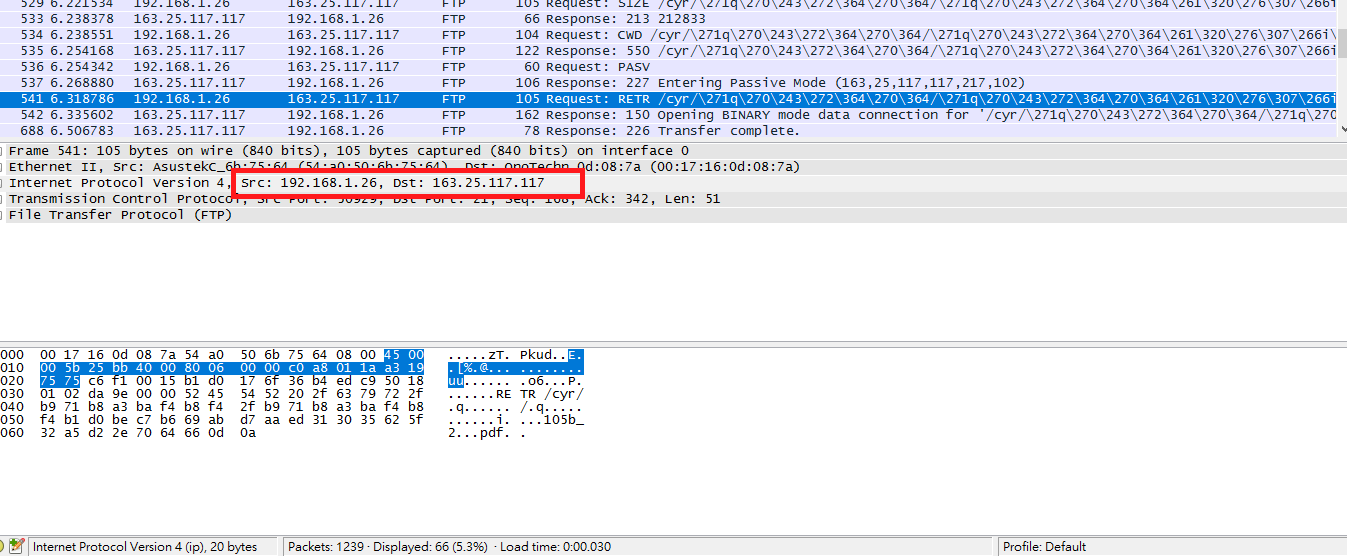
組員:資管二甲 B0444146 林子由

一、IP



->首先我們到<ftp://163.25.117.117/cyr/>電腦網路，下載來一個大約208KB的pdf檔。

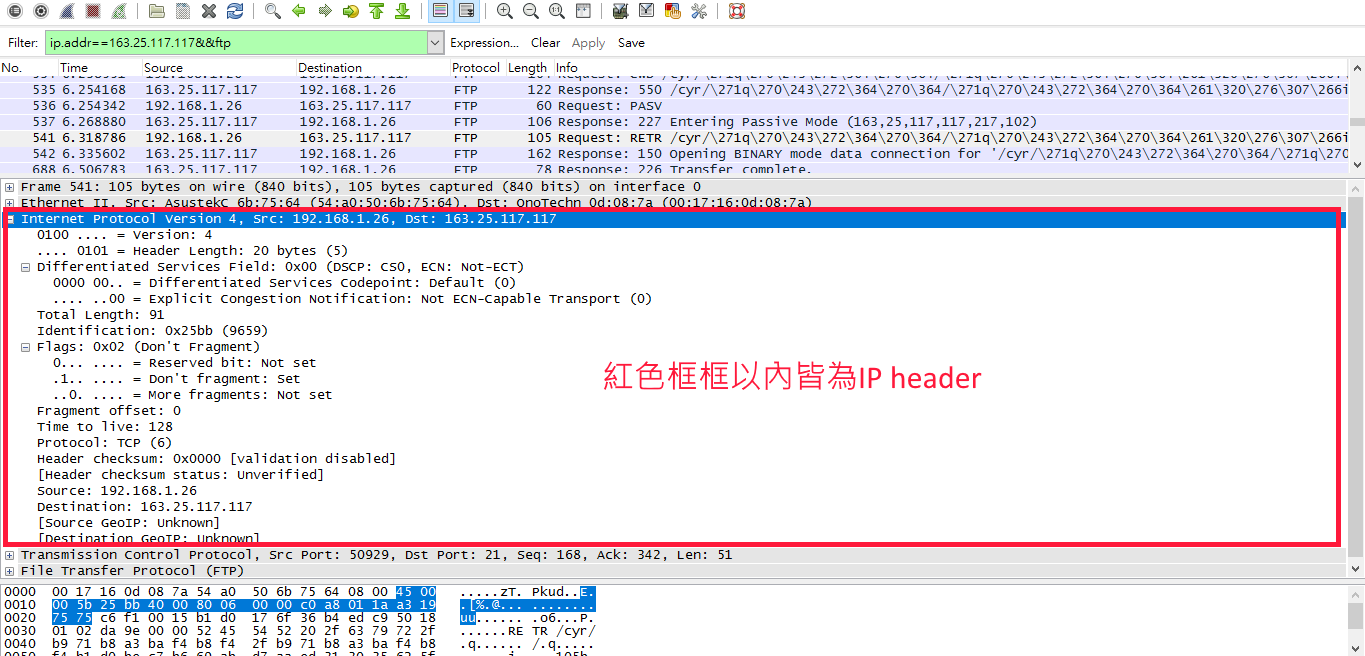
1.指出你的主機與伺服器之IP位址。



◎主機IP位址:192.168.1.26

◎伺服器IP位址:163.25.117.117

2.任意選取一個IP封包，說明IP標頭的各個欄位。



1.Version: 4(代表IPv4)，宣告這個 IP 封包的版本，例如目前慣用的還是 IPv4 這個版本就在這裡宣告。

2.Header Length: 因為IPv4封包有可能包含不同數量的選用欄位(包含於IPv4的封包標頭中)，所以我們需要這4個位元來判斷IP封包中真正開始存放資料的位置為何?大多數IP封包並不含選用欄位，因此一般的IP封包標頭長度為20位元組。

3.Differentiated Services Codepoint(簡稱DSCP):

最初定義為[服務類型](https://en.wikipedia.org/wiki/Type_of_service)（Type of service）欄位。此欄位現在被定義[RFC 2474](https://tools.ietf.org/html/rfc2474) [區分服務](https://en.wikipedia.org/wiki/Differentiated_services)（DiffServ的）。新技術正在出現，需要實時數據流，因此利用了DSCP領域。一個最具代表性的例子是[IP語音](https://en.wikipedia.org/wiki/Voice_over_IP)（VoIP），用於交互式數據語音交換。

4.Explicit Congestion Notification(簡稱ECN):

該欄位在[RFC 3168](https://tools.ietf.org/html/rfc3168)中定義，允許端到端通知[網絡擁塞](https://en.wikipedia.org/wiki/Network_congestion)而不丟棄數據包。ECN是一個可選功能，僅在兩個端點都支持並且願意使用它時才使用。它僅在底層網絡支持時才有效。

**DSCP+ECN就是Type of service**

5.Total Length:此欄位代表IP封包的總長度(標頭加上資料)，單位為位元組。因為此一欄位的長度為16位元，所以IP封包長度的理論上限為65535位元組。然而，封包很少會大於1500個位元組。

6.Identification:

由於 IP 封包必須要放在 MAC 當中。不過，如果 IP 封包太大的話，就得先要將 IP 再重組成較小的封包然後再放到 MAC 當中。而當 IP 被重組時，每個來自同一 IP 的小封包就得要有個識別碼以告知接收端這些小封包其實是來自同一個 IP 封包才行。 也就是說，假如 IP 封包其實是 65536 位元組那麼大 (前一個 Total Length 有規定)， 那麼這個 IP 就得要再被分成更小的 IP 分段後才能塞進 MAC 訊框中。那麼每個小 IP 分段是否來自同一 IP 資料，那就是這個識別碼的功用。

7.Flags: 010(代表不可分段且為最後分段)

這個地方的內容為『0DM』，其意義為：  
D：若為 0 表示可以分段，若為 1 表示不可分段  
M：若為 0 表示此 IP 為最後分段，若為 1 表示非最後分段。

8.Fragment offset: 這個13位元欄位指明了每個分段相對於原始封包開頭的偏移量，以8位元組作單位。

9.Time to live(簡稱TTL):

這個8位元欄位避免封包在網際網路中永遠存在（例如陷入路由環路）。存活時間以秒為單位，但小於一秒的時間均向上取整數到一秒。在現實中，這實際上成了一個跳數計數器：封包經過的每路由器都將此欄位減1，當此欄位等於0時，封包不再向下一跳傳送並被丟棄，最大值是255。常規地，一份[ICMP](https://zh.wikipedia.org/wiki/ICMP)封包被發收回封包傳送端說明其傳送的封包已被丟棄。這也是[traceroute](https://zh.wikipedia.org/wiki/Traceroute)的核心原理。

10.Protocol:6(代表資料部分要轉交給TCP)

只有當IP封包抵達最終目的時，才會使用到這個欄位。這個欄位的數值會指出這份IP封包的資料部分，應該要轉交給哪個特定的傳輸層協定。

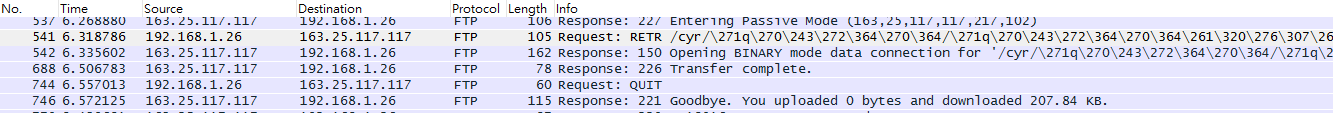
*1: ICMP 2: IGMP 3: GGP 4: IP 6: TCP 8: EGP 17: UDP*

11.Header checksum: 用來檢查這個 IP 表頭的錯誤檢驗之用。

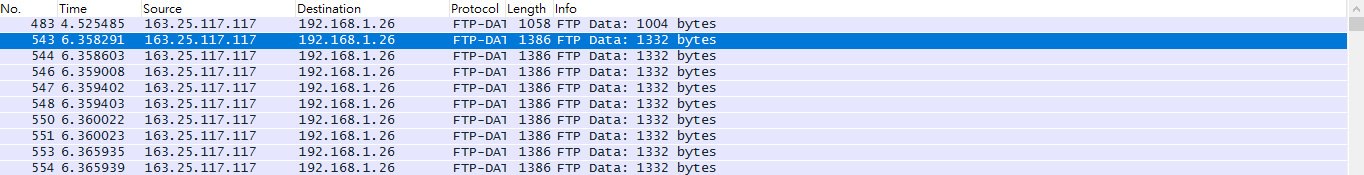
12.Source:192.168.1.26來源端IP位址。

13.Destination:163.25.117.117目的端IP位址。

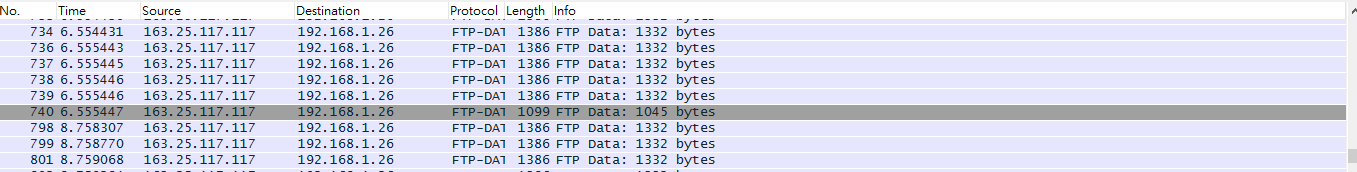
3. 說明所擷取IP封包的數量，並指出每個封包的大小。



->從No.541封包開始(由RETR得知，RETR代表開始從FTP下載資料)~No.744封包(由QUIT得知結束)。其餘的封包為ftp-data如下圖所示，從No.543~No.740



…



由No.543和No.740的封包識別碼為5811和5970得知(皆為不分段封包)，ftp-data的封包總共5970-5811+1=160個，並加上ftp封包4個。

=>所以在下載過程中，總共擷取到164個IP封包

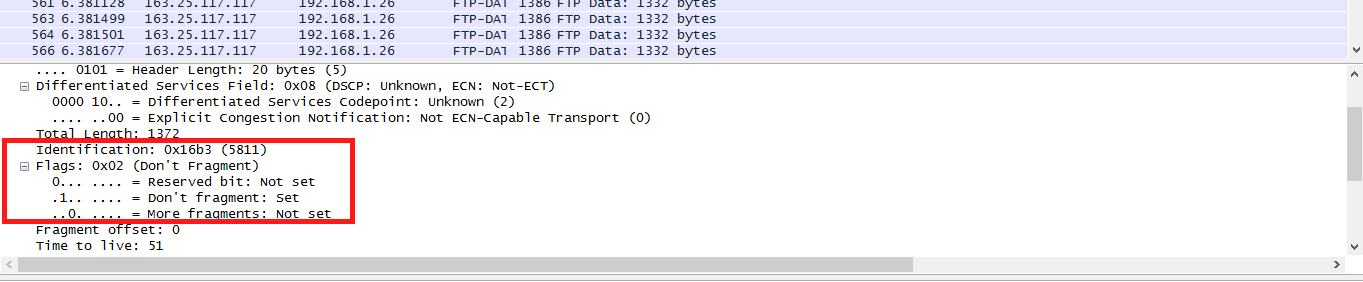
除了4個下ftp指令的封包大小較不一樣外，其他傳輸檔案資料的封包大小為1386bytes，但扣除一些標頭欄位(54bytes)後皆為1332bytes，以及最後一筆傳送檔案資料的封包大小為1099-54=1045bytes。



我們可以經計算ftp-data封包的大小來比對是否為檔案大小。

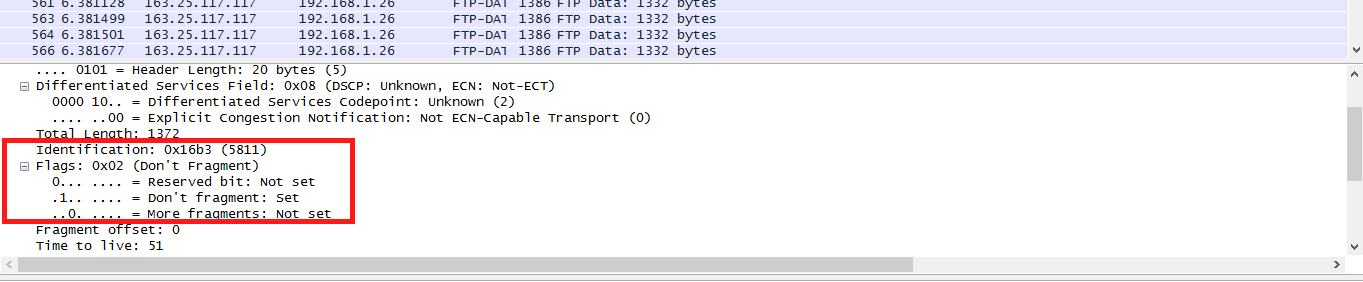
(160-1) \*1332+1\*1045=211788+1045=212833

4.觀察IP標頭中Identification欄位的變化。



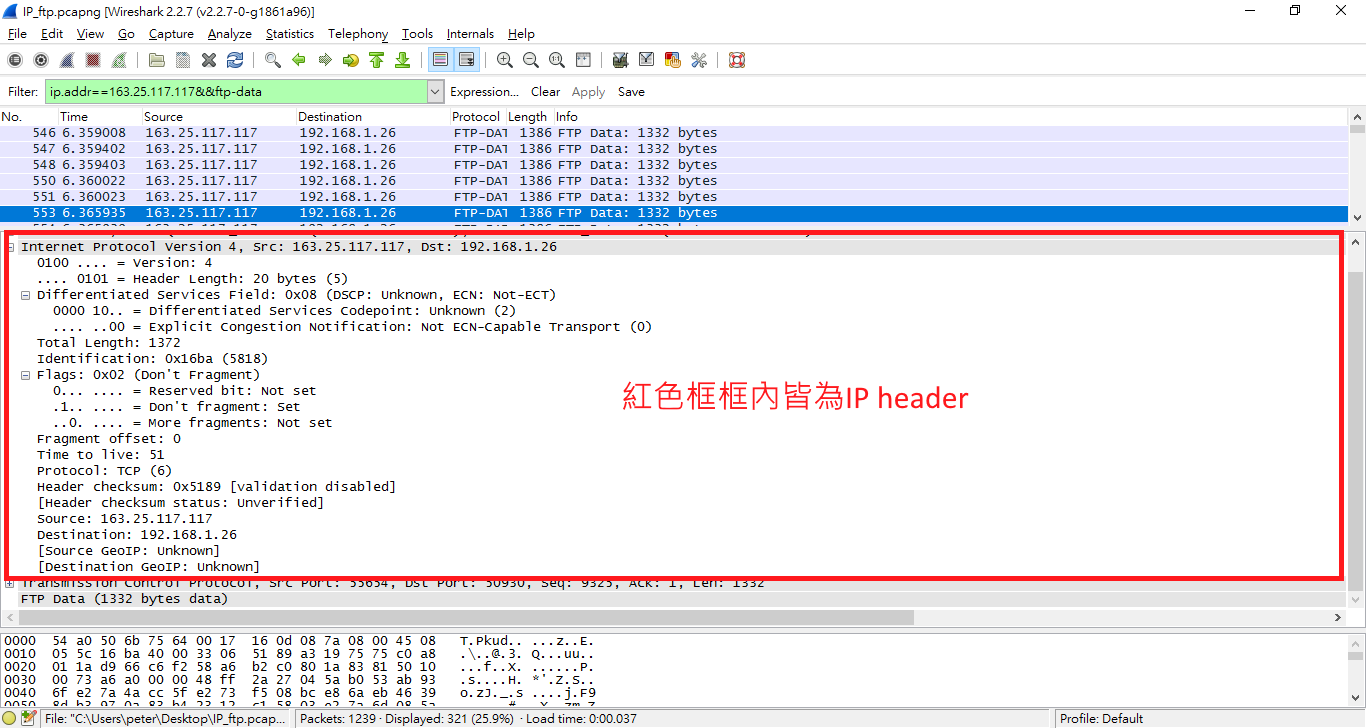
經由觀察每個封包的Flag欄位得知所有的封包皆為不分段封包，所以Identification欄位會從最一開始的5811，一直+1到最後的5970。

5.觀察是否有被分割成小塊 (Fragment) 的IP封包。



一樣經由觀察每個封包的Flag欄位中的第二個數值(皆為1)，得知所有的IP封包皆為不分段IP封包，因此並沒有被分割成小塊的IP封包。

6.觀察一連串檔案下載的封包中，哪些IP標頭欄位的值會改變，哪些維持不變。



經由一連串的觀察

Version、Header Length、DSCP、ECN、Flags、Fragment offset、TTL、Protocol、Source、Destination的值皆不會改變，皆如上圖所示。

Total Length、Identification、Header checksum的值則會改變:

Total Length除了最後一個封包為1085外，其餘的皆為1372

Identification的值每個封包會依序+1

Header checksum的值則會一直改變

二、心得

這是最後一次的Wireshark作業，雖然四次的作業從頭到尾都只有我在做，但是也讓我對Wireshark更加的熟悉，像是對一些篩選條件的語法都更加的熟練，而這次的作業是任意選一個檔案伺服器抓取IP封包，並利用IP封包中的header去判斷一些問題，我選取了FTP Server來做這次的擷取封包。經過這次的實作後，我對從很久以前就知道的FTP和IPv4，有更加多元地了解，像是FTP每個運作的步驟，例如:從一開的anonymous登入進去，或是用輸入密碼的登入，還有List要求伺服器傳回遠端目錄下所有的檔案列表，一直到最後RETR真正下載到最後的pdf檔，讓以前只知道FTP是個檔案伺服器的我有更深入地了解到FTP是如何運作的；至於IPv4，從國小的時候就一直有聽爸媽說過，但我卻一直不知道是什麼，以至於每次自己的IP都不會設，經由這次的實作後，雖然說還沒有到完全了解的地步，但是也讓我從以前幾乎不知道的地步，成長了一些，其碼知道了每個IP header代表甚麼意思，以及封包是否有分段。