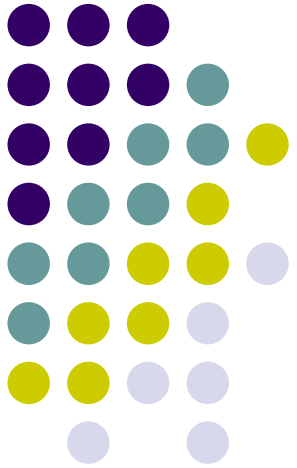
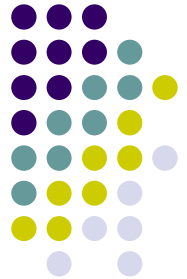
實驗八 TCP協定分析



實驗目的

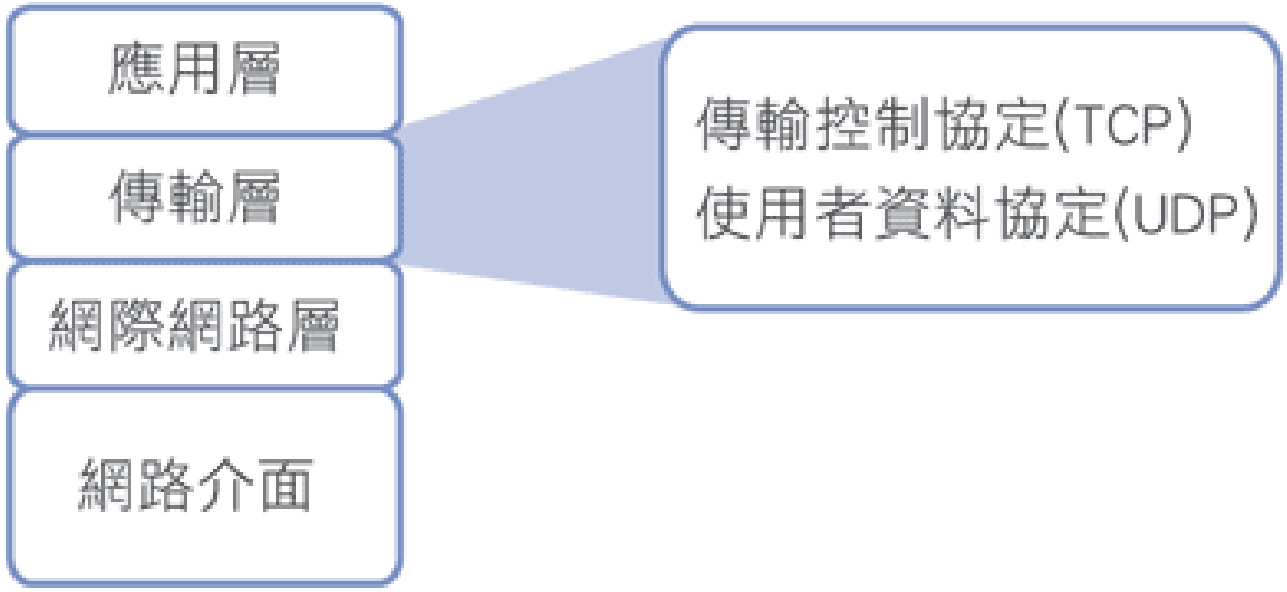
- 明瞭可靠傳輸層的基礎觀念
- TCP協定下區段資料傳送的格式

背景資料



- TCP協定主要是為了在主機間實現高可靠性的封包交換傳輸協定，因為TCP協定主要是用在網路不可靠的時候完成通信，對於軍方可能特別有用，但是以目前來說，對於政府部門以及商業單位也非常適合。
- TCP點對點的可靠協定，它支援許多種的網路應用程式，TCP對於下層的服務並沒有多少要求，它直接假設下層只能提供不可靠的資料傳輸服務，而且可以在多種硬體介面組成的網路上使用。

TCP協定



TCP協定

- TCP的上面一層就是應用層，下面是IP協定，上層包括一系列類似作業系統中中斷的運用，對於上層應用程式來說，TCP應該可以同步的來傳送資料，下層假設為IP協定。
- 為了在不可靠的網路介面上建立可靠的傳送資料服務，TCP必須解決可靠性，流量控制的問題，而且必須能為上層的應用程式來提供多個埠口，用來同時為多個應用程式提供資料，同時，TCP必須解決連接的問題，這樣子的話，TCP才是個可靠的通訊協定，而最後也要克服通訊安全的問題。

TCP協定

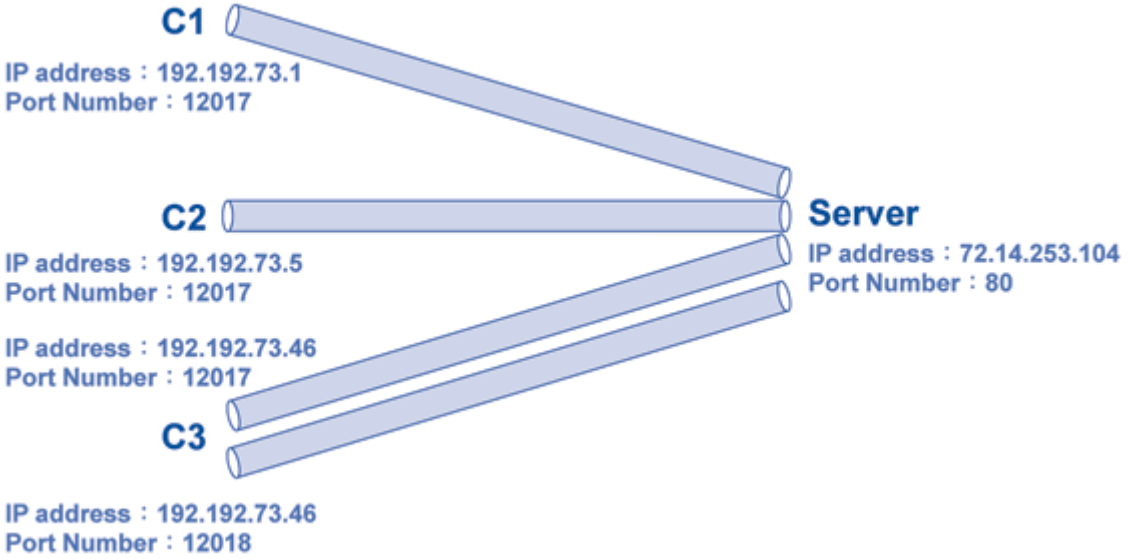
- 在運行TCP協定的電腦主機上，TCP可以被看成是一個模組，和文件系統區別不大，TCP也可以調用一些作業系統的功能，TCP不直接和網路相連，控制網路的任務是由專門的設備來完成，TCP協定只是用來配置埠號，IP協定向TCP協定提供所有TCP需求的服務。
- TCP連接是可靠的，而且保証了傳送區段的順序，保証順序的能力是用一個序號來達成的，區段內也包括一個序列號，表示接收方準備好這個序號的區段，在TCP協定傳送一個區段時，它會同時把這個區段放入重新發送的序列中，同時啟動計數器，如果收到了確認訊息，就會把重送的區段序列刪除，如果超過計數時間時，就會將這個區段重送。

TCP協定

- 每一個用於傳送TCP資料的通道都有一個埠號記錄，因這個記錄是由每個TCP的終端來確定的，因此TCP可能是不唯一，為了保証這個數值的唯一性，要使用網路位址和埠號的組合來達到唯一的目的，稱這個為Socket。
- 一個通訊由連接兩端的Socket來標示，本地端的Socket可能和不同的外部Socket來通訊，這種通訊是全雙工的。

TCP 連線

- 所有TCP的傳輸都必須在TCP連線(TCP Connection)下進行，TCP連線的定義至少需要四組參數，即來源端的IP地址、連線埠號和目的端的IP地址、連線埠號。在下圖中雖有三個客戶端機器連線至伺服器端，C1和C2雖連線埠號相同但IP地址不同故互為獨立的TCP連線，C3中連線埠號兩組故也互為獨立的TCP連線，所以圖中有4組TCP連線。

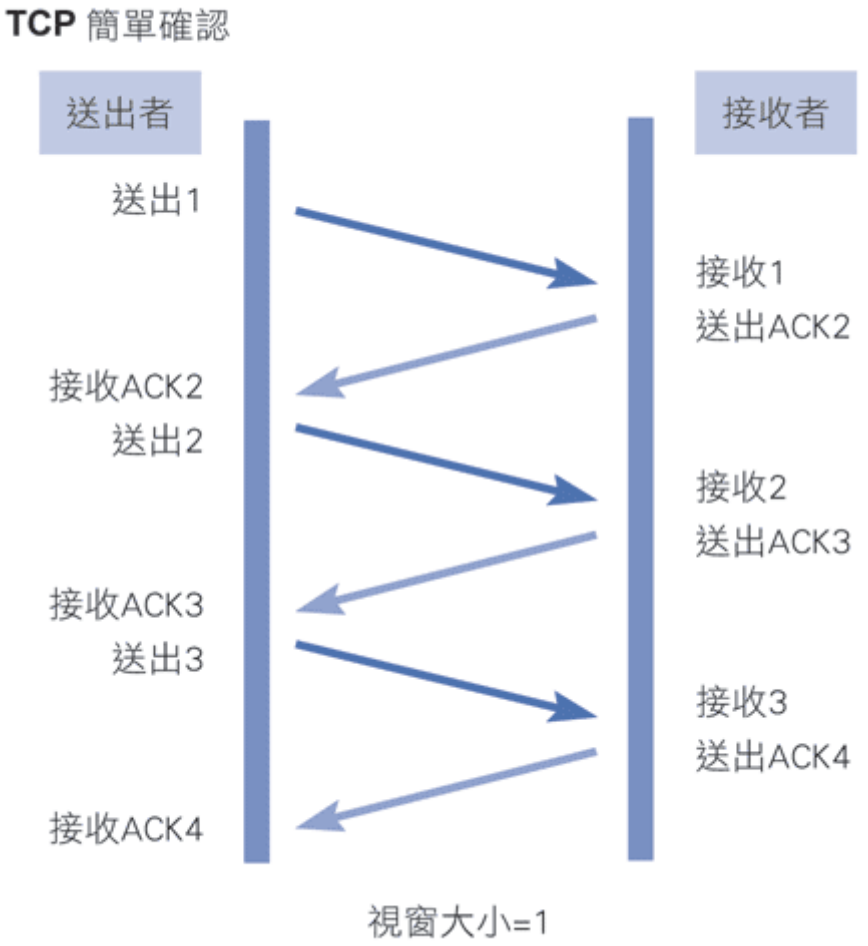


確認與重送

- 肯定確認與重新傳輸（Positive Acknowledgement and Retransmission；PAR），是許多通訊協定用來確保可信度，經常使用的一種技術。有了PAR，發送端送出一個封包後，會開始計時，待對方確認後，才會再送出下一個封包。若尚未收到確認，而計時器已逾時，則發送端會再傳輸一次該封包，並重新計時。上述傳送的過程，雖然具有確認與重送的功能， 但在效能方面會造成很大的問題，當發送端每傳送出去一個封包後，便只能等待，一直等到收到對方確認封包後，才能傳送下一個封包，所以在整個傳送過程中，絕大部份時間都浪費在等待確認封包。這種機制只有在早期通訊品質和計算機效能均很差的年代出現過，如X.25，現今已不復存在。
- 由於TCP具有PAR，所以UDP可能可為單向式傳輸，TCP必為雙向式傳輸，不過傳輸時上下載頻寬不必相同(如ADSL)，上下載路徑也可以不相同(如單向式Cable Modem)。

確認與重送

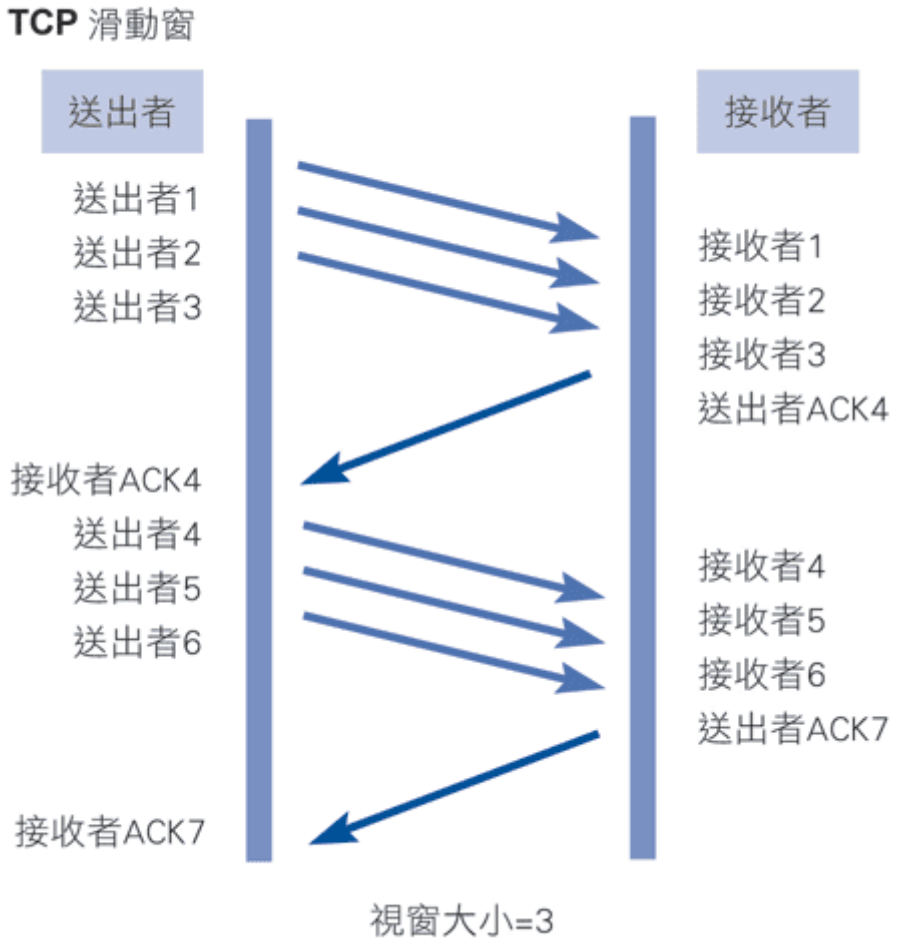
- 視窗大小決定在收到目的地確認之前，一次可以傳送的資料量。代表視窗大小的數字越大（位元組），主機可以傳輸的資料量就越多。當主機傳輸視窗大小數目的位元組資料後，就必須等收到確認以後，才可以再傳下面的訊息。例如，若視窗的大小為1，則傳完每個(1)區段後，都必須經過確認，才可以再傳下一個區段。



緩衝區與
滑動視窗（Sliding Window）

- 滑動視窗是一種資料流量控制技巧，它要求發送端設備在傳送一定量的資料以後，必須接收對方確認。
- 想像有一扇窗戶，如果將窗戶向左拉開時，空隙增大此時空氣進來的多，將窗戶向右關閉時，空隙變小此時空氣進來的少，所以利用窗戶的位置決定空氣進來的數量，水龍頭左轉右轉也相同道理。不過由於計算機處理資訊流僅有一個方向，並不能像上述的左拉右關左轉右轉，所以此時將窗戶的外框不再是固定而是可以滑動，如此外框和窗戶的相對位置便可決定空隙大小決定流量。
- 例如，若視窗大小為3，就表示發送端設備可以連續送出三個封包給接收端。然後就必須等待對方確認。如果目的地接收到三個位元組，就會傳送確認到來源設備，來源設備就可以再傳送另外三個位元組。若由於某種原因，接收端沒有收到三個位元組（例如，可能是因緩衝區溢位），就不送出確認訊息。因為來源沒有收到確認，它就會知道必須重新傳送位元組，並且要降低傳送速率。

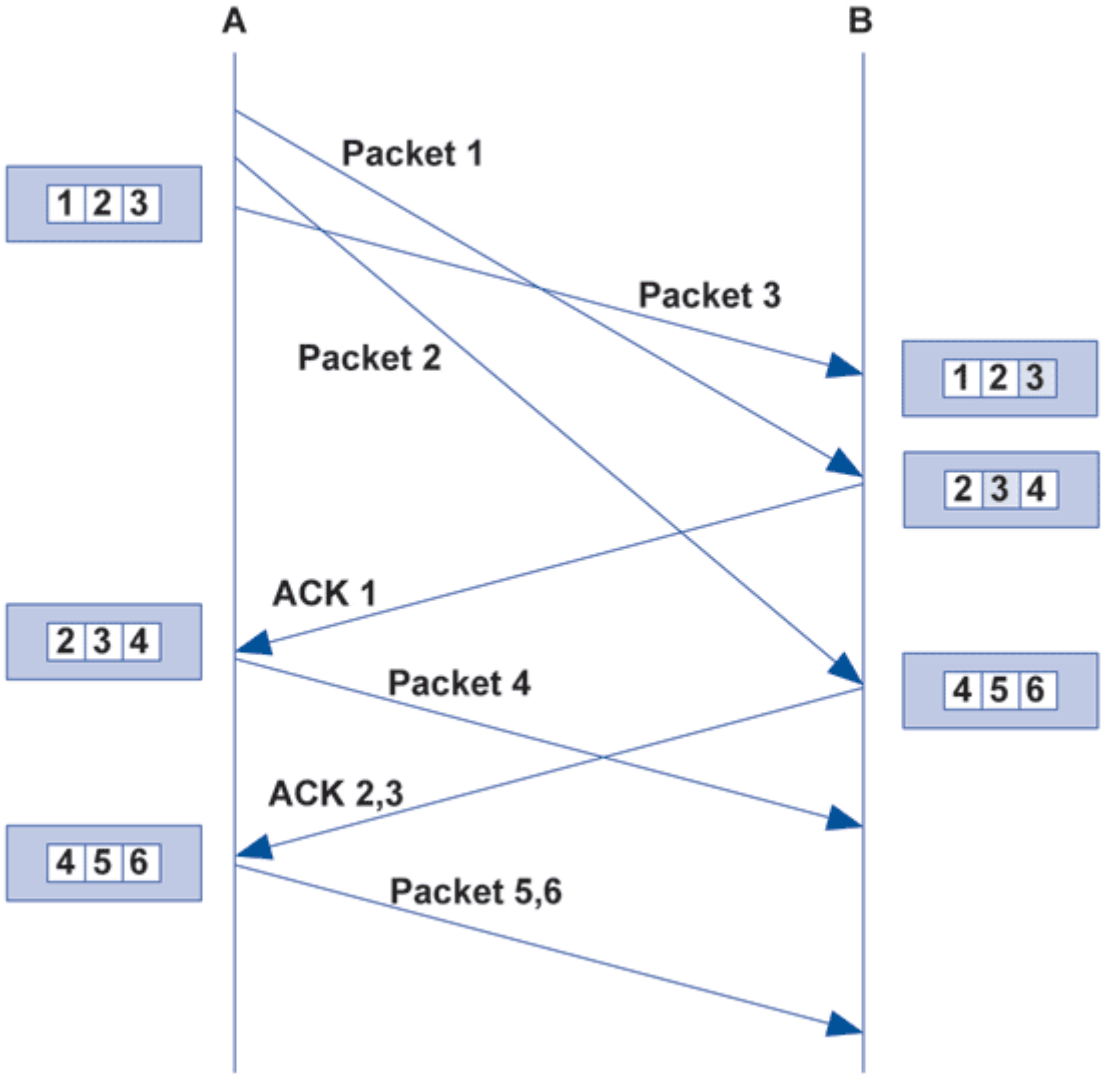
緩衝區與
滑動視窗（Sliding Window）



緩衝區與
滑動視窗（Sliding Window）

- TCP能夠視網路流量情況需要，隨時調整資料傳送速度，流量控制主要是靠滑動視窗的大小來調整，當Window Size變小時，流量也會變小，當Window Size變大時，流量也會變大，相對地，較大的Window Size會耗費較多的電腦資源。
- 決定Window Size的大小是相當複雜的過程，不過我們需知道的是Window Size是由目的端決定，而且會依網路流量調整並非一成不變的。
- 由於目的端需用緩衝區來暫存封包，所以可以保證資料到達順序，如圖中雖然packet 3先到達，但此時並不送用至應用層而是暫存起來，等到packet 1、packet 2到達，packet 3才用至應用層。

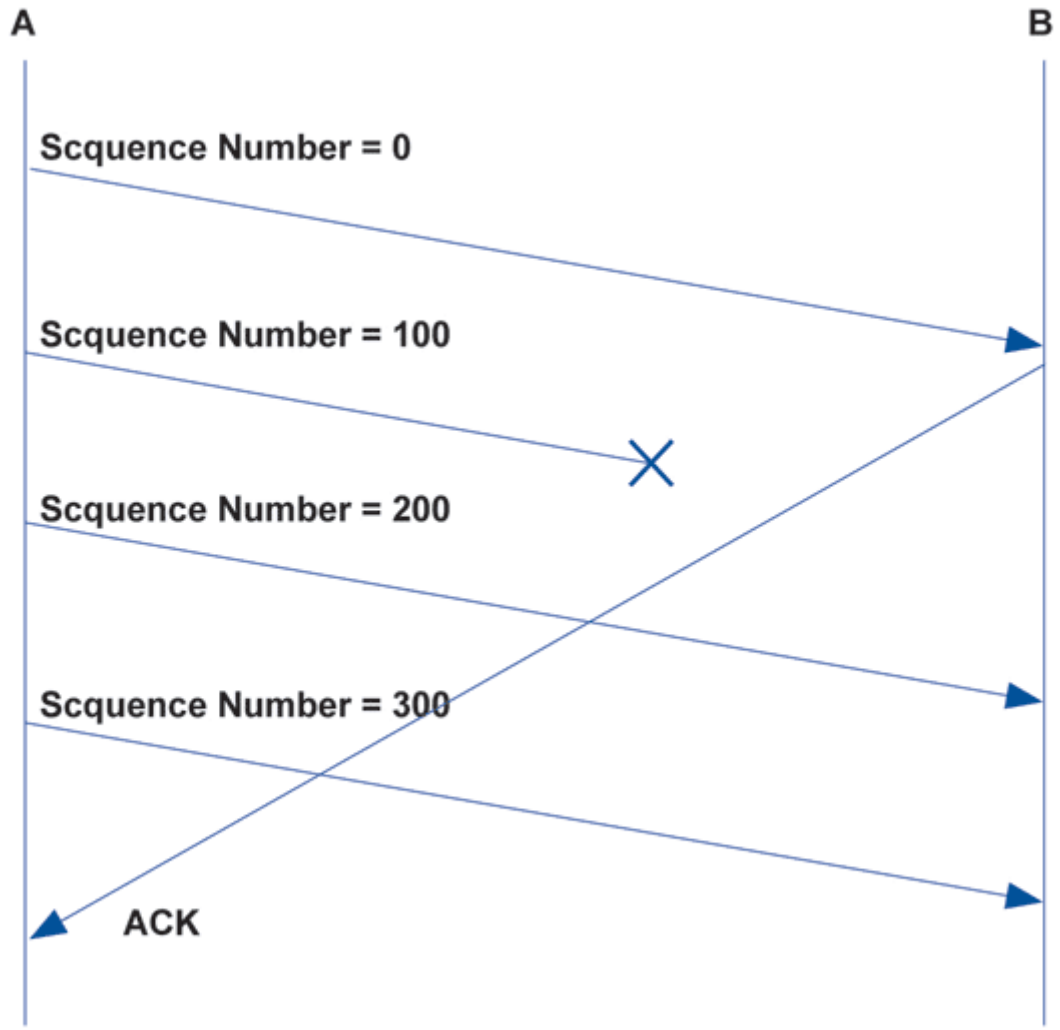
緩衝區與
滑動視窗（Sliding Window）



緩衝區與滑動視窗（Sliding Window）

- 當封包傳送發生錯誤時，滑動視窗有兩種重送解決機制，回到-N(go-back-N)和選擇性重送(selective repeat) 。
- 如圖為例，當[Sequence Number =100]封包傳送失敗時，如果採用回到-N的重送機制，則指標回到[Sequence Number =100]封包開始重送，機制實施較容易，但是效能較差，因為[Sequence Number =200]和[Sequence Number =300]還要送一次，Window Size越大效能越差。如果採用選擇性重送的重送機制，則只重送[Sequence Number =100]區段，機制實施較困難，但是效能較佳。
- 早期的TCP是採用回到-N的重送機制，而目前大多改採用選擇性重送機制，所以在建立連線時雙方會互相溝通是否要使用SACK-Permitted (Selective Acknowledgement)功能。

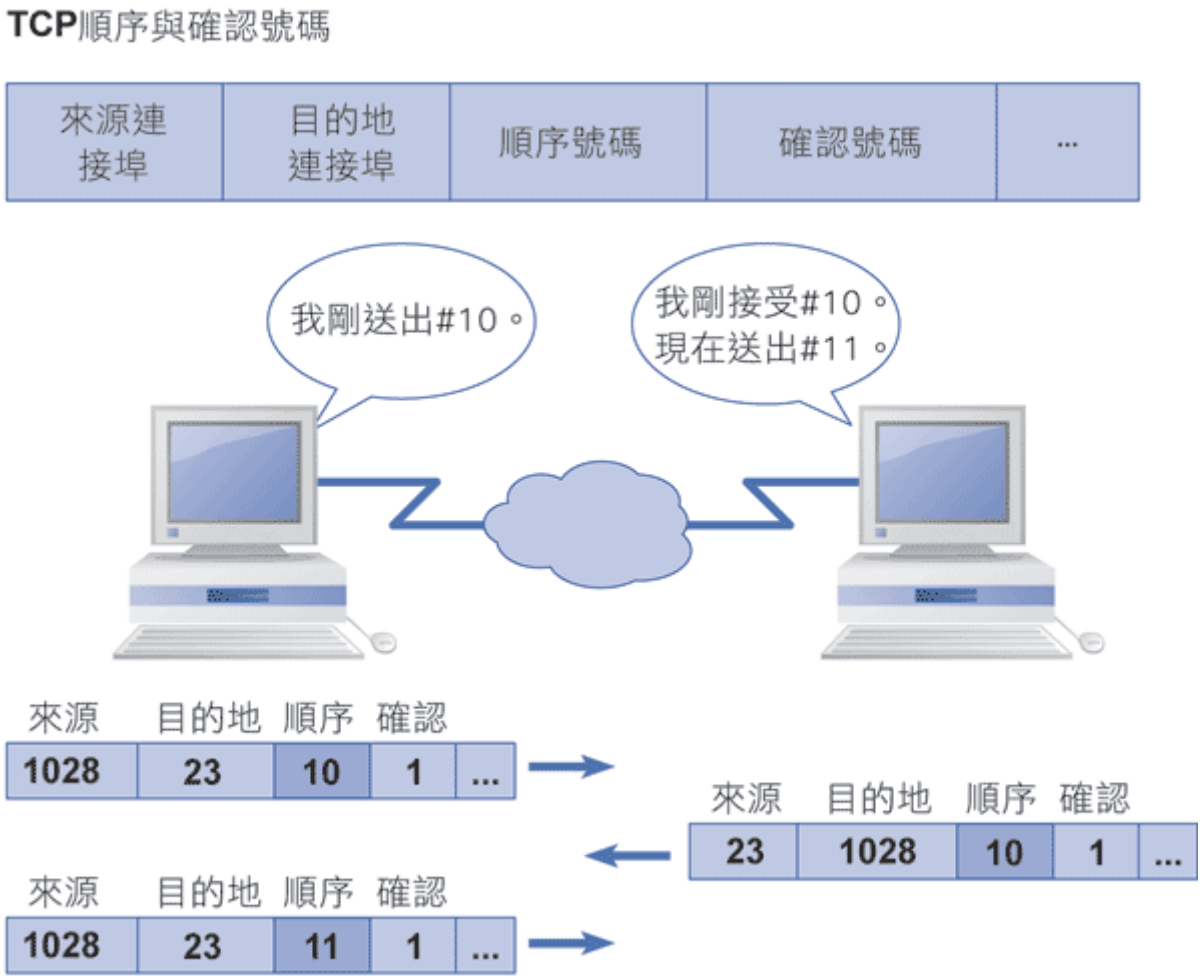
緩衝區與
滑動視窗（Sliding Window）



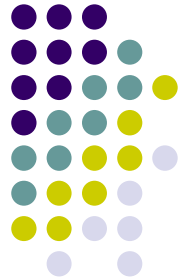
Sequence Number

- TCP以向前參照確認來提供區段排序作業，每個資料在傳輸前都會先加以編號，到目的地工作站時，TCP會將這些區段重新組成完整的訊息。若資料中少了某個順序號碼，就會重傳該區段。區段傳出後，若在一段特定時間內沒有收到確認，也會要求重傳。

Sequence Number



建立連線

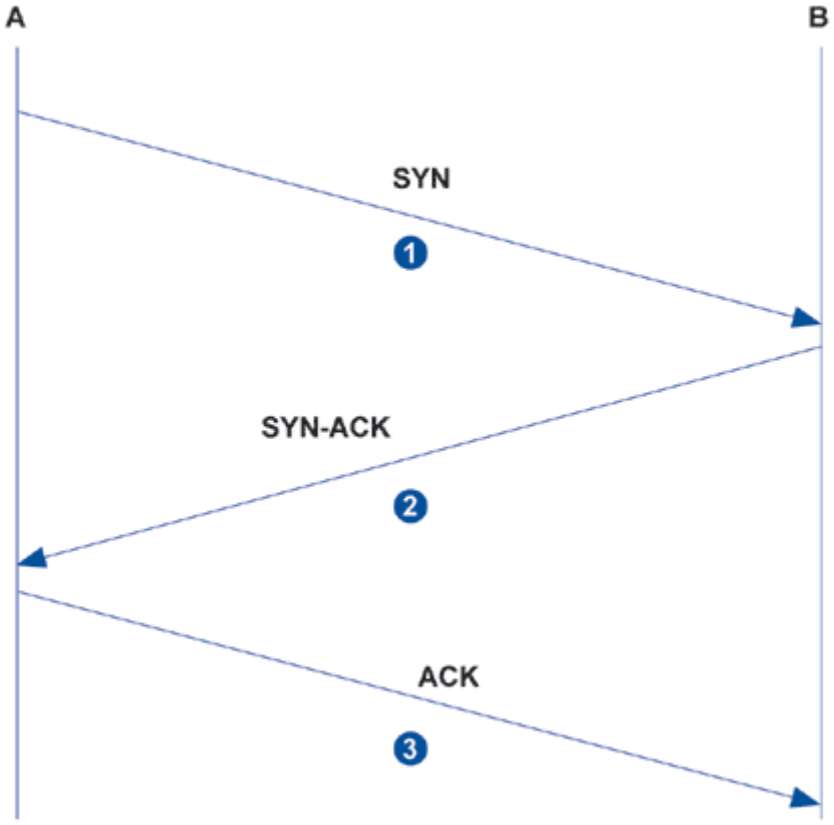


- TCP連線採用的是連線導向式(connection-oriented)，請注意並非連線交換(connection switches)，也就是說在連線建立階段，不會決定一條來源與目的地之間的路徑，且將資源加以保留。而僅僅是連線建立時和連線中，讓雙方知道對方使用的各項TCP參數，如Initial Sequence Number、Window Size、支援的TCP選擇性功能等。
- TCP主機之間使用三向式交握法（Three-way Handshake），建立連接導向式的會談，三向式交握法在資料抵達終端之前，先將兩終端連線同步化，這種在連接順序時的介紹性順序編號交換非常重要。它可確保由於傳輸問題導致遺失的任何資料，都可以在稍後復原。

建立連線--三向式交握法

- 第1步驟：A(客戶端)送出SYN封包給B(伺服端)，此封包包含A、B雙方的連接埠號、A的Initial Sequence Number(ISNA)、TCP選擇性功能(如MSS、SACK-Permitted)。
- 第2步驟：B在收到SYN封包，回覆SYN-ACK封包，此封包包含B的ISNB、Acknowledge Number ACK(ISNA+1)、Window Size，用來控制A的Send Window大小。
- 第3步驟：A在收到SYN-ACK封包後，發出 ACK 封包，此封包包含Sequence Number(ISNA+1)、Acknowledge Number ACK(ISNB+1)、Window Size A的Receive Window大小。

建立連線--三向式交握法



阻斷服務攻擊 (DOS:Denial-of-service attack)

- 假設A向B傳送SYN封包後突然當機或離線，那麼B在發出SYN-ACK封包是無法收到A的ACK封包（第三次握手無法完成），由於網路是以善意為原則所以這種情況下B一般會重試（再次發送SYN-ACK給用戶端）並等待一段時間後放棄這個未完成的連線，這段時間的長度我們稱為SYN Timeout。
- ，一般來說這個時間是分鐘的數量級（大約為30秒-2分鐘），一個用戶出現異常導致伺服器的一個行程等待並不是什麼很大的問題，但如果有一個惡意的攻擊者大量使用此情況，伺服器端將為了維護一個非常大的半連線列表而消耗非常多的CPU時間和記憶體資源，何況還要不斷對這個列表中的IP進行SYN-ACK的重試。

阻斷服務攻擊 (DOS:Denial-of-service attack)

- 如果伺服器的網路作業系統不夠強壯，最後的結果往往是堆疊溢位崩潰，即使伺服器端的系統足夠強大，伺服器端也將忙於處理攻擊者偽造的TCP連接請求而無暇理睬客戶的正常請求，此時從正常客戶的角度看來，伺服器失去反應，這種情況即為伺服器端受到了SYN Flood攻擊。
- 網路上有針對SYN Flood攻擊的種種防禦之道，但大多會拖累伺服器的效能，並不易取捨，即使可防禦單一客戶端的SYN Flood攻擊，但如果同時有多部(一般是被植入後門)對伺服器進行SYN Flood攻擊，這是分散式阻斷服務攻擊(DDoS:Distributed Denial-of-Service)，要防禦並不容易。

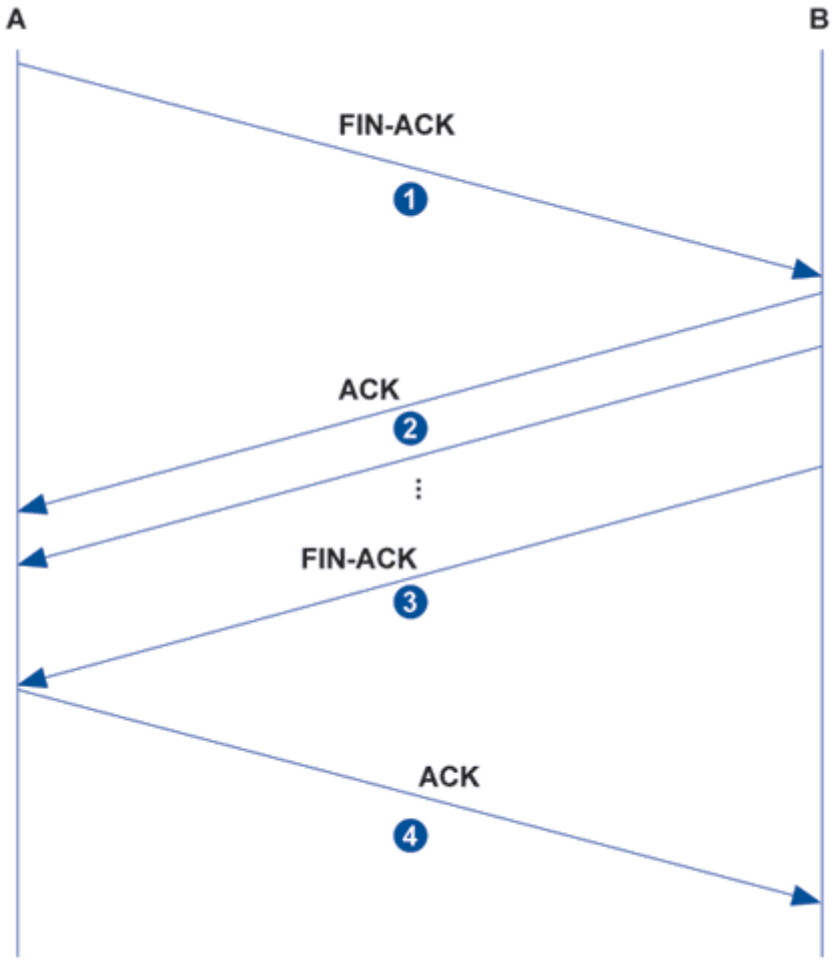
TCP選擇性功能

- TCP常見的選擇性功能有MSS (Maximum Segment Size)、SACK-Permitted和SACK。
- MSS此選項主要是在連線建立時，用來指定所能傳送TCP Payload的最大長度，所以在連線過程中，雙方每次傳送的TCP Payload長度都不會大於該值，所以系統可以適當的設定此值，使得在IP層在傳輸時不會產生封包需被切割的情況，所以我們是不太可能會擷取到有分割的IP封包。

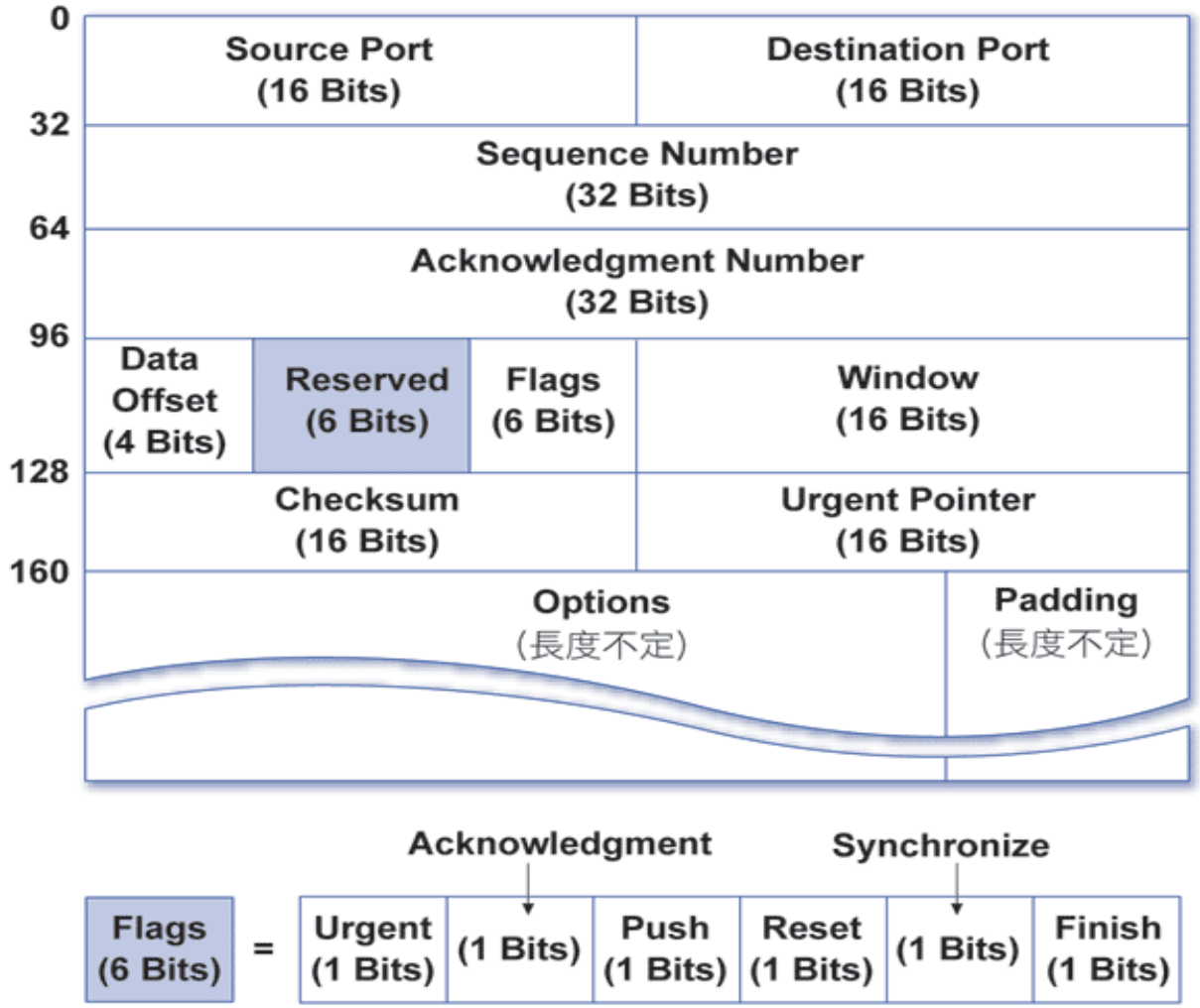
中止連線

- TCP 連線若要中止，為將連線所用的資源（連接埠、記憶體等等）釋放出來，必須經由特定的連線中止步驟，雖然建立連線時可區分為主動端與被動端，但是雙方都可以主動提出中止連線的要求，這也造成某些惡意攻擊程式會從中中止雙方的連線，連線中止的過程有4個步驟。
  - 第1步驟：A送出FIN-ACK 封包給B。
  - 第2步驟：為處理FIN-ACK 封包之前的送出的資料，B送出 ACK 封包給 A。。
  - 第3步驟：B送出 FIN-ACK 封包給 A。
  - 第4步驟：A 送出 ACK 封包給 B。

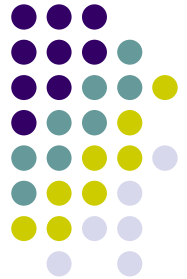
中止連線



TCP表頭欄位


0
32
64
96
128
160
Source Port (16 Bits)
Destination Port (16 Bits)
Sequence Number (32 Bits)
Acknowledgment Number (32 Bits)
Data Offset (4 Bits)
Reserved (6 Bits)
Flags (6 Bits)
Window (16 Bits)
Checksum (16 Bits)
Urgent Pointer (16 Bits)
Options (長度不定)
Padding (長度不定)
Acknowledgment
Synchronize
Flags (6 Bits)
=
Urgent (1 Bits)
(1 Bits)
Push (1 Bits)
Reset (1 Bits)
(1 Bits)
Finish (1 Bits)

TCP表頭欄位

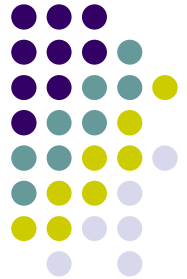


- Source Port（來源連接埠號）：長度為2 Bytes，記錄來源端上層應用程式所用的TCP連接埠號。
- Destination Port（目的連接埠號）：長度為2 Bytes，記錄目的端上層應用程式所用的TCP連接埠號。
- Sequence Number（序號）：長度為4 Bytes，記錄傳送資料過程的累積序號，當來源主機開始發送訊號到目的主機時，系統會內定一個初始值作為序號的開始；傳送封包過程中，序號會隨著傳送完成的Byte數，依次將序號循環疊加上去，因此藉由序列號碼欄位值，便可以判定所傳送的資料是否已經收到。

TCP表頭欄位

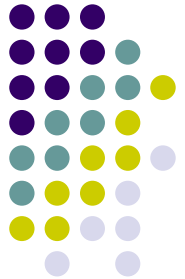
- Acknowledge Number（回應號碼）：長度為4 Bytes，與序列號碼欄位配合進行訊息傳遞確認工作。目的主機在收到來源主機的資料後，會將收到的序列號碼加上傳送資料的Byte長度，將之填入回應號碼欄位，傳回給發送端驗證。
- Data Offset（資料長度）：長度為4 Bits，記錄TCP表頭的長度，欄位值的單位為4 Bytes。例如此欄位值是5，代表TCP表頭的長度為20 Bytes。
- Reserved（保留）：長度為6 Bits，保留用途，設為0。

TCP表頭欄位



- Flags（特殊用途位元）：長度為6 Bits，每個Bit可代表TCP封包的一種Flag。共有6種Flag：
  - Urgent（緊急）：設為1時，表示此封包為緊急資料包，需要立即處理。
  - Acknowledge（回應）：設為1時，表示此封包帶有回應確認訊息。
  - Push（推進）：設為1時，表示要求封包直接將攜帶的資料往上層應用程式送去，訊息不再經過TCP的封包處理，或等待緩衝區完全接收資料後再行處理。
  - Reset（重設）：設為1時，可立即中斷TCP連線，並重新對TCP封包進行設定傳送。
  - Synchronize（同步）：設為1時，表示TCP正在進行雙方同步溝通。每次TCP連線建立之初，都必須執行這項同步溝通的工作。
  - Finish（結束）：設為1時，表示某端點的TCP資料傳送工作已經結束，來源主機與目的主機都要有正確的回英才表示是個完整的結束。

TCP表頭欄位



- Window（視窗大小）：長度為2 Bytes，主要是紀錄資料封包在傳送或接收後，傳送主機內的緩衝區還剩下多少可以裝載封包的資料空間，可用來控制流量。
- Checksum（檢查碼）：長度為2 Bytes，記錄錯誤檢查碼，運作方式與UDP的錯誤檢查碼相同。
- Urgent Pointer（緊急資料指標）：長度為2 Bytes，當Urgent Flag設為1時，接到此封包的主機都必須優先對封包進行處理。本欄位記錄TCP資料中，屬於Urgent資料的最後一個Byte。
- Options與Padding：
  - Options欄位長度不定，可用來擴充TCP的功能。
  - Padding欄位是為了讓TCP表頭（包含Options欄位）剛好是4 Bytes的倍數。

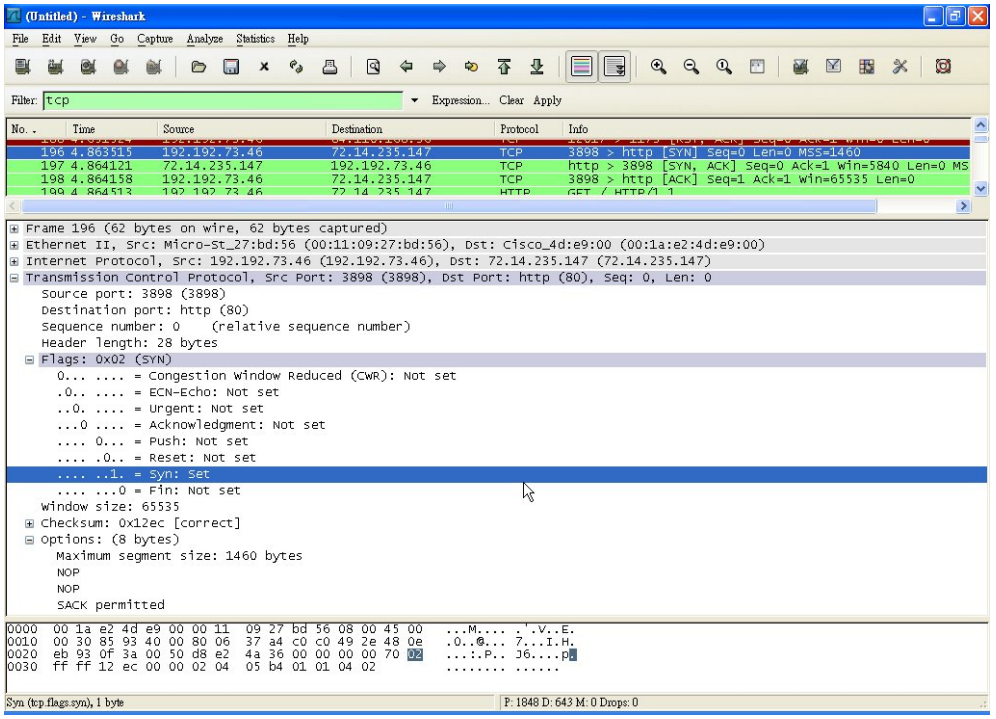
UDP和TCP的比較

- UDP和TCP協定的主要分別，是在於兩者在如何作到資料傳送的可靠性的方式不同。TCP協定中包含了專門的傳送確認機制，當接收方收到傳送方傳來的封包時，會送出一個確認的訊息給傳送方，傳送方在收到接收方傳來的確認訊息後才會繼續傳送資料，若沒有收到確認的訊息時，就會一直等待直到收到確認訊息為止。
- 和TCP協定不同的是，UDP協定並沒有提供資料傳送的保証機制。如果在傳送的過程中出現任何的資料毀損的話，那麼協定本身是不會像傳送方做出任何的通知的，因此，通常我們都把UDP協定稱為不可靠的傳輸協定。相對於TCP協定，UDP協定的另外一個相異之處在於如何接收突發性的多個資料封包，不同於TCP協定，UDP並不能確保資料封包的發送和接收順序。

實驗方法--TCP建立連線封包

說明：

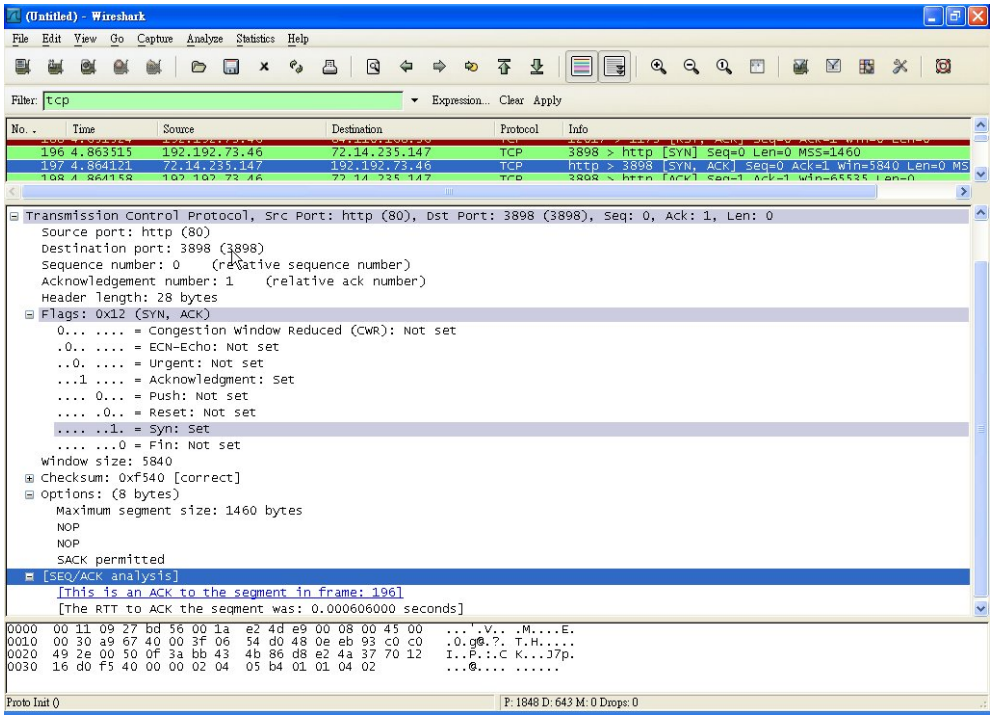
- 1.Source port = 3898；
- 2.Destination port = http (80)；
- 3.Sequence Number = 0xd8e24a36，相對為0；
- 4.Acknowledgement Number，此時系統不使用；
- 5.Header Length = 28 Bytes；
- 6.Flags = 只有SYN設為1；
- 7.Window Size = 65535；
- 8.Checksum = 0x12ec；
- 9.Options:(8 bytes)
- Maximum Segment Size: 1460 bytes;
- SACK permitted



實驗方法--TCP建立連線封包

說明：

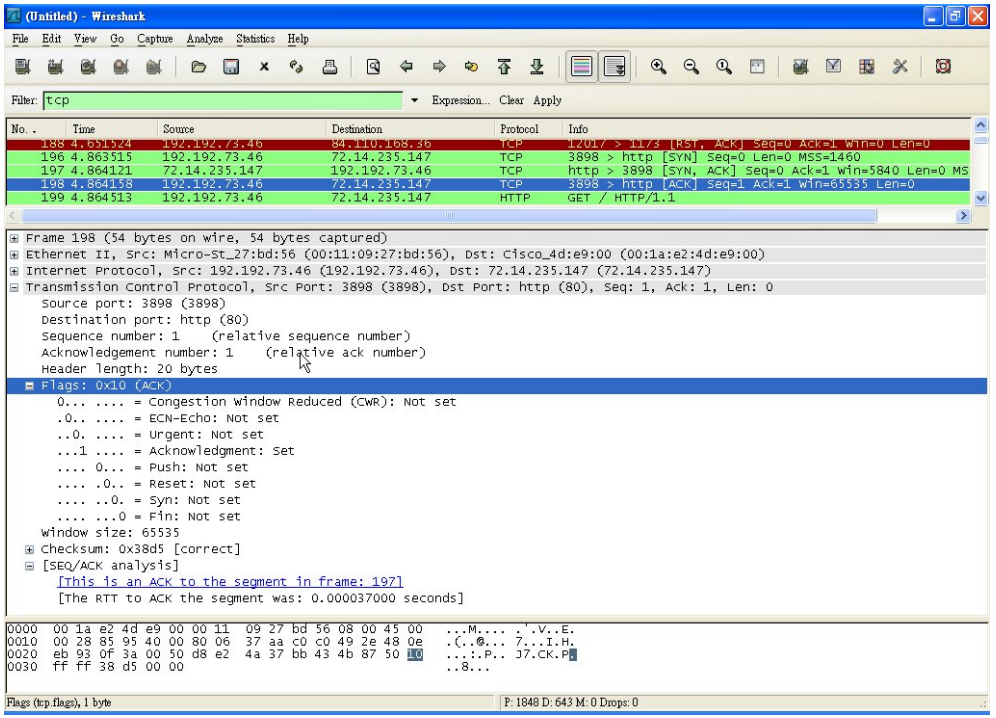
- 1.Source port = http (80)；
- 2.Destination port = 3898；
- 3.Sequence Number = 0xbb434686，相對為0；
- 4.Acknowledgement Number = 0xd8e24a37，相對為1；
- 5.Header Length = 28 Bytes；
- 6.Flags = SYN, ACK設為1；
- 7.Window Size = 5840；
- 8.Checksum = 0xf540；
- 9.Options:(8 bytes)
- Maximum Segment Size: 1460 bytes;
- SACK permitted



實驗方法--TCP建立連線封包

說明：

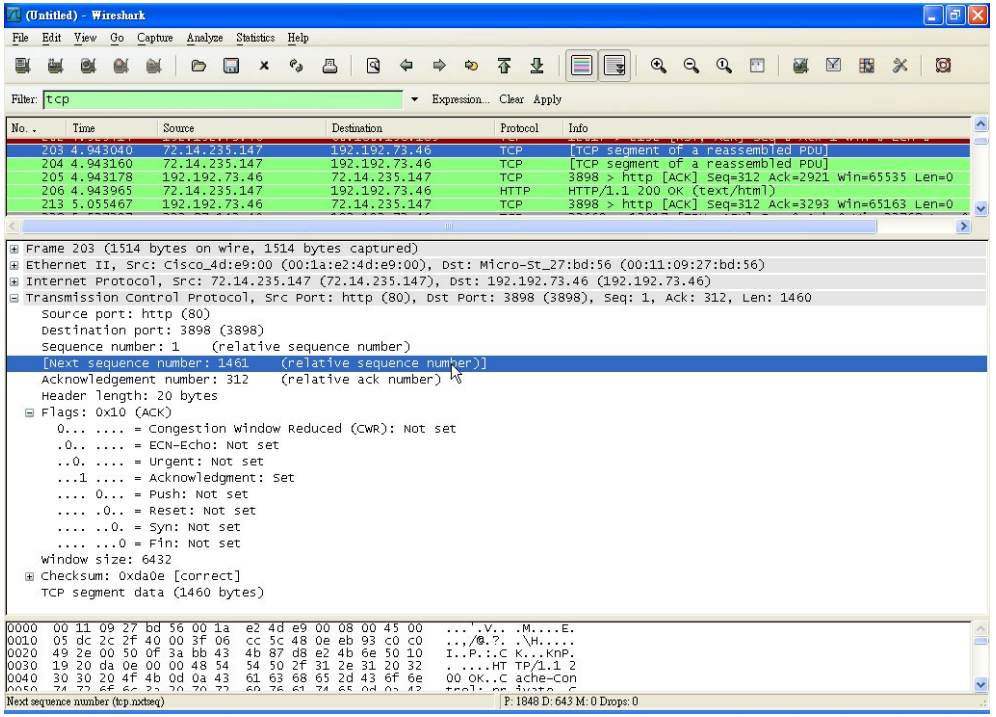
- 1.Source port = 3898；
- 2.Destination port = http (80)；
- 3.Sequence Number = 0xd8e24a37，相對為1；
- 4.Acknowledgement Number = 0xbb434687，相對為1；
- 5.Header Length = 20 Bytes；
- 6.Flags = 只有ACK設為1；
- 7.Window Size = 65535；
- 8.Checksum = 0x38d5；



實驗方法--TCP區段傳送

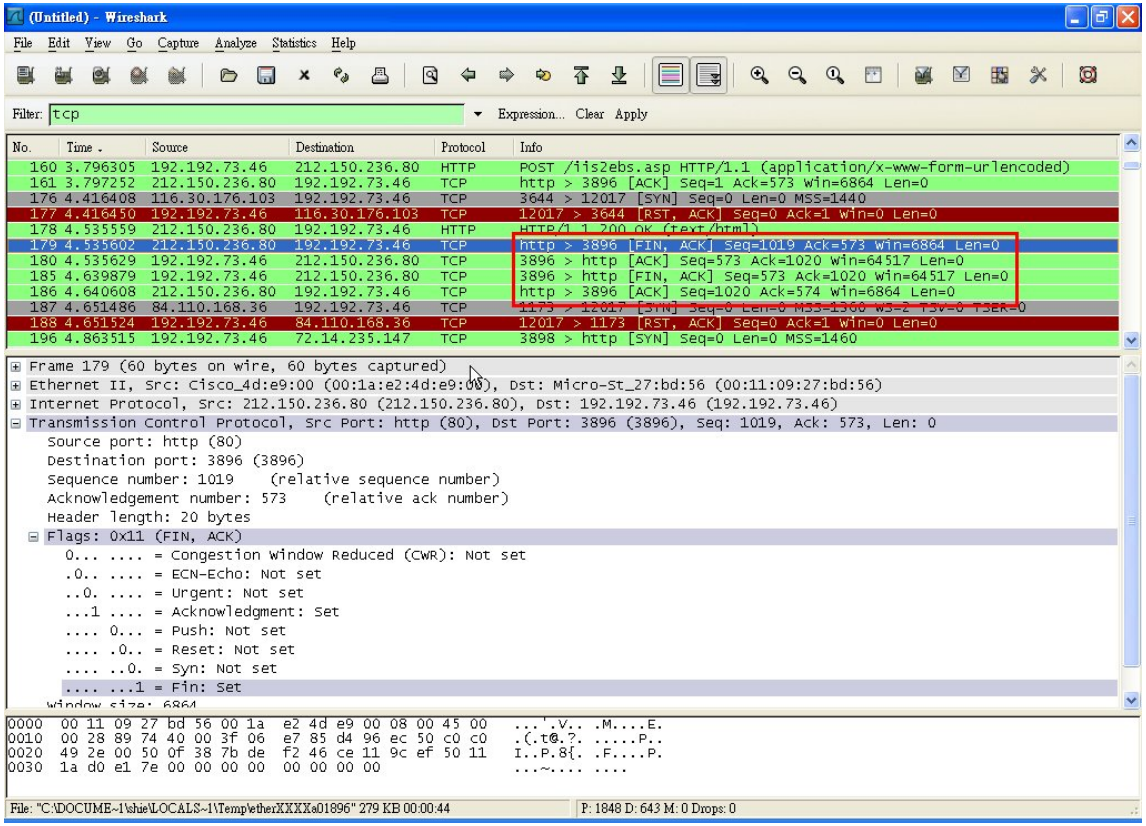
說明：

- 1.Source port = 3898；
- 2.Destination port = http (80)；
- 3.Sequence Number = 0xd8e24a37，相對為1；
- 4.Acknowledgement Number = 0xbb434687，相對為1；
- 5.Header Length = 20 Bytes；
- 6.Flags = 只有ACK設為1；
- 7.Window Size = 6432；
- 8.Checksum = 0xda0e；



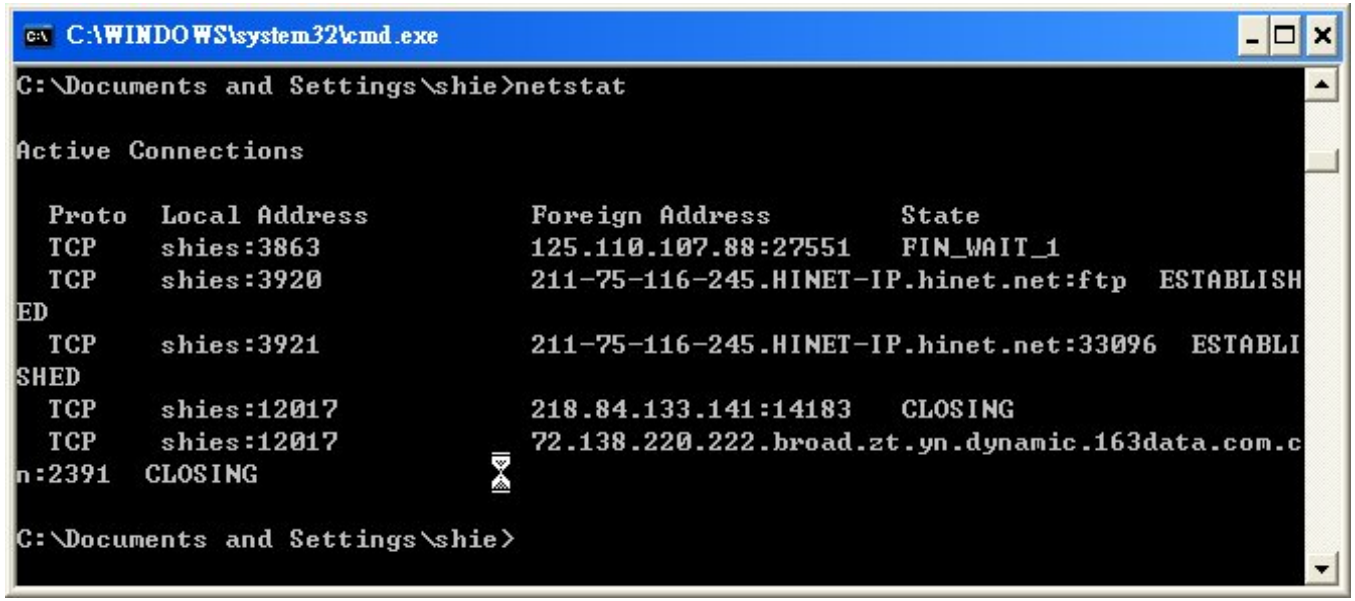
實驗方法--TCP中止連線封包

- 由圖中可看到TCP中止連線的四步驟封包。



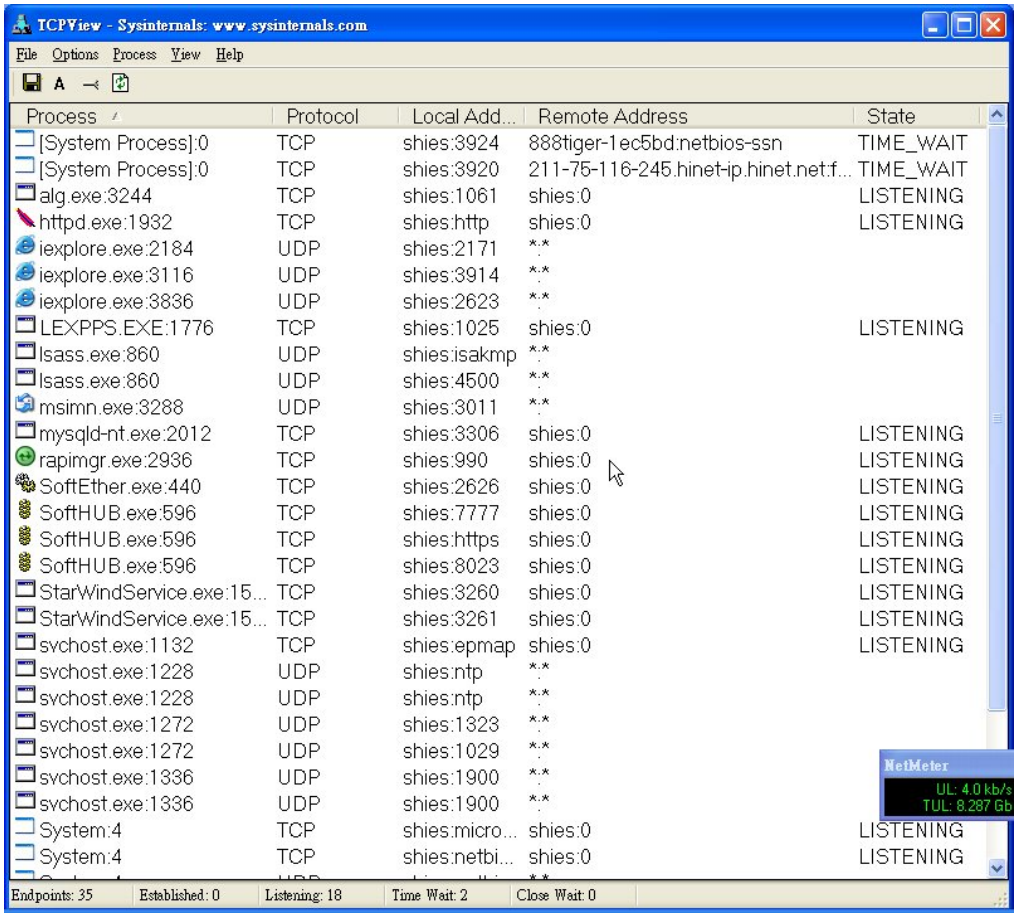
觀察系統連線情況

- netstat指令



觀察系統連線情況

- TCPView



學習評量

1. 何謂滑動視窗？有哪些網路架構使用到它？
2. 繪出TCP協定的狀態圖。
3. 何謂Silly window syndrome？
4. 說明TCP協定如何利用window大小控制擁塞？
5. 說明TCP協定如何如何計算RTT？
6. 說明TCP協定中旗號之作用？並用wireshark實際擷取區段。
7. 說明TCP協定中潛在的風險？