

UDP Source Port Steganography

100321003 賴原群、100321048 李東岳

資訊隱藏 (Steganography) 的目的與加密不同。資訊隱藏的目的是傳送秘密訊息時不被第三方察覺；加密的目的是讓傳送秘密訊息不能被第三方解讀，但容易第三方被察覺。在應用層傳輸秘密訊息通常會加密，但有心人士攔截到後，很容易察覺是否經過加密，之後便可想盡各種方法進行破解。由於傳輸的訊息都是放在應用層，因此如果能把秘密訊息隱藏在較不受人關注的底層 Header，便可以達到資訊隱藏的目的，瞞過有心人士的耳目。

傳統資料隱藏是將秘密資訊藏在媒體資料裏面，在應用層裡進行藏密動作。這邊我們進行了不一樣的做法，利用傳輸層沒用到的空間來進行藏密的動作，然後在應用層裡定義 header 來進行排序的動作。以下依依介紹傳輸層以及應用層的隱藏流程：

傳輸層：

使用 UDP 來進行資料隱藏，將資料藏在 source port 裡面。由於 UDP header 比較簡單，處理上比較方便，相對於 connection oriented 的 TCP，UDP 是 connectionless，在頻繁更改 source port 的情況下，TCP 需要不斷重新進行 three-way handshake 建立連線，效能會比較差。而選擇 source port 的原因是 Source port 的值一般是隨機產生，相對其他欄位，當值改變時較不會被懷疑。Source port 有兩個 byte 的空間，我們將前 2 個 bit 作為提供後面所藏入的資訊，後 12 個 bit 作為資料藏入的地方。中間兩個未用到的 bit 保留給未來使用。

前兩 bit	代表意義
00	藏入結束
01	後 4 bit LSB
10	後 8 bit LSB
11	後 12 bit LSB

為盡量提高藏密度使用 1.5 byte (12bits) 藏密，以 byte 為單位的資料隱藏後，會餘下 0, 4, 8 三種可能的 bit 數 ($(n * 8) \% 12 = 0, 4, 8$)，因此需要指示藏入多少 bits。由於 UDP 並不保證傳送完後接收方所接收到的順序是對的，因此需要有個方法來確保機密資訊切成小部份經由 source port 偷渡過後，能夠以正確的順序還原回來，我們利用了應用層來進行排序的動作。

應用層：

保留	Timestamp	Seq	Ack	message
0	1~8	9~10	11~12	13~end

上表的單位為byte，也就是說使用者輸入完訊息後我們會在前面定義這 12 個byte為header，利用此header來進行排序的動作。Timestamp 紀錄了使用者輸入機密訊息時的當下時間，因此如果輸入了第二次機密訊息，可以利用timestamp來進行辨認。Seq為sequence number，因為會將機密訊息切成好幾個片段，每個片段紀錄了seq讓接收端接收時能夠依照seq來進行排列的動作。ack功能尚未完成，未來保有發展空間。

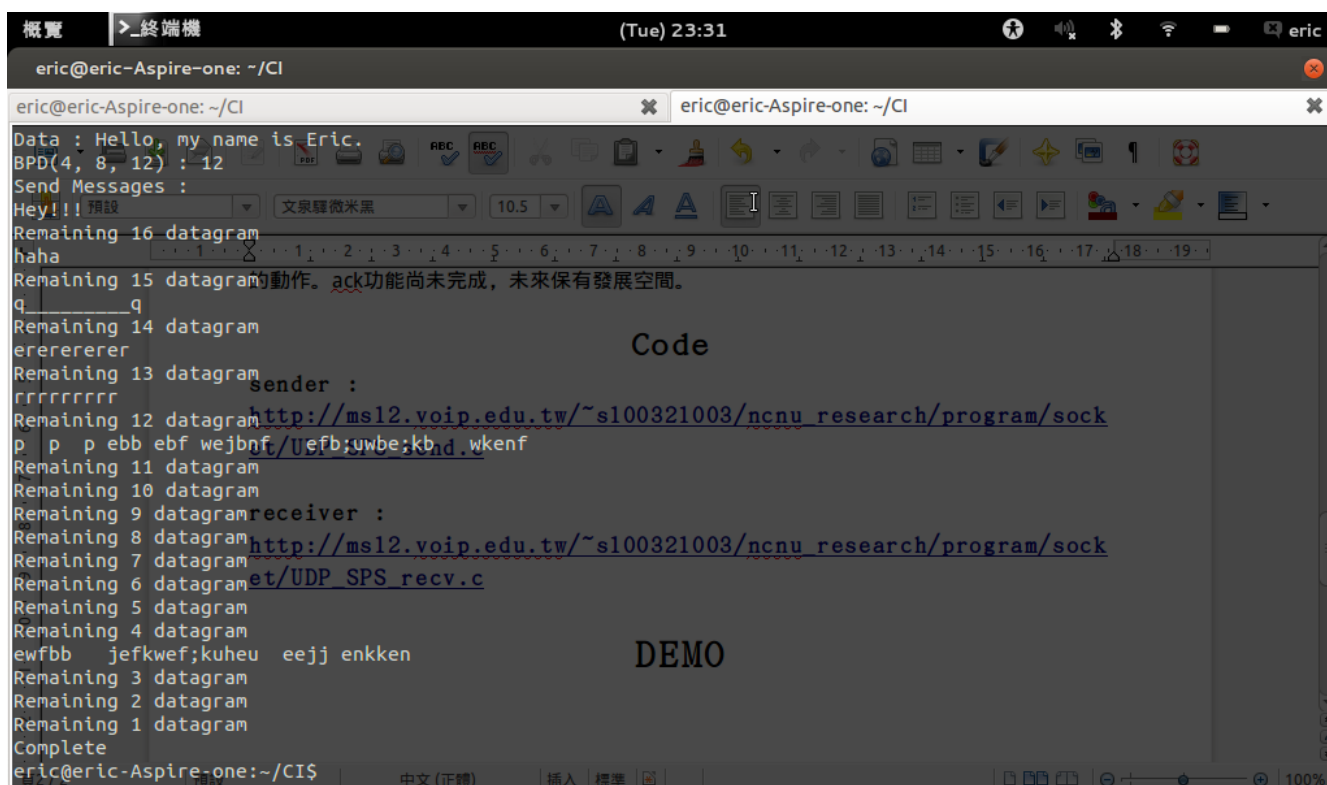
Code

sender : http://ms12.voip.edu.tw/~s100321003/ncnu_research/program/socket/UDP_SPS_send.c

receiver : http://ms12.voip.edu.tw/~s100321003/ncnu_research/program/socket/UDP_SPS_rcv.c

DEMO

sender :



```
eric@eric-Aspire-one: ~/CI
Data : Hello, my name is Eric.
BPD(4, 8, 12) : 12
Send Messages :
Hey!!!
Remaining 16 datagram
haha
Remaining 15 datagram
Remaining 14 datagram
Remaining 13 datagram
Remaining 12 datagram
Remaining 11 datagram
Remaining 10 datagram
Remaining 9 datagram
Remaining 8 datagram
Remaining 7 datagram
Remaining 6 datagram
Remaining 5 datagram
Remaining 4 datagram
Remaining 3 datagram
Remaining 2 datagram
Remaining 1 datagram
Complete
eric@eric-Aspire-one:~/CI$
```

receiver :

```
eric@eric-Aspire-one: ~/CI
eric@eric-Aspire-one: ~/CI$ ./new_recv.exe 5000
Hey!!!
:haha
:q
:ererer
:
: p
: p
: p
: ebb
: ebf
: wejbnf
: efb;uwe;kb
: wkenf
: ewfbb
: jefkwef;kuheu
: eejj
: enkken
secret message: Hello, my name is Eric.
Code
sender :
http://msl2.voip.edu.tw/~s100321003/ncnu_research/program/socket/UDP_SPS_send.c
receiver :
http://msl2.voip.edu.tw/~s100321003/ncnu_research/program/socket/UDP_SPS_recv.c
DEMO
```

Wireshark (Wed) 00:06

Capturing from wlan0 [Wireshark 1.6.7]

Filter: udp.dstport == 5000

No.	Time	Source	Destination	Protocol	Length	Info
701	433.802406	192.168.137.140	10.21.21.171	UDP	56	Source port: 50854 Destination port: 5000
702	435.346148	192.168.137.140	10.21.21.171	UDP	56	Source port: 51818 Destination port: 5000

Frame 701: 56 bytes on wire (448 bits), 56 bytes captured (448 bits)

Ethernet II, Src: HonHaiPr_21:cb:93 (00:26:5e:21:cb:93), Dst: 16:2f:68:39:e2:ab (16:2f:68:39:e2:ab)

Internet Protocol Version 4, Src: 192.168.137.140 (192.168.137.140), Dst: 10.21.21.171 (10.21.21.171)

User Datagram Protocol, Src Port: 50854 (50854), Dst Port: 5000 (5000)

Source port: 50854 (50854)

Destination port: 5000 (5000)

Length: 22

Checksum: 0xf8e3 [validation disabled]

Data (14 bytes)

Data: 00a7b4041197fc040000000000073

[Length: 14]

```
0010  00 2a 36 6d 40 00 00 11 9a 61 c0 a8 89 8c 0a 15  .*m@. .a.....
0020  15 ab c6 a6 13 88 00 16 f8 e3 00 a7 b4 04 11 97  .....s
0030  fc 04 00 00 00 00 00 73
```