

Introduction to Computer Network Proj 2 Report

0416324 胡安鳳

0.環境

OS: Linux Ubuntu 16.04 LTS x64

網卡: 使用以下指令

```
alfons@alfons:~|⇒ lspci | egrep -i --color 'network|ethernet'
02:00.0 Ethernet controller: Qualcomm Atheros AR8161 Gigabit Ethernet (rev 10)
03:00.0 Network controller: Intel Corporation Wireless 7260 (rev bb)
```

Browser:Firefox & Chrome

IP:192.168.0.110

1.

(1) 連上短網頁與長網頁均發送了一個 HTTP GET REQUEST

```
11 5.383133294 192.168.0.100 140.113.168.116 TCP 74 41334 → 80 [SYN] Seq=0 Win=29200 Len=0
12 5.384682387 140.113.168.116 192.168.0.100 TCP 74 80 → 41334 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
13 5.384693816 192.168.0.100 140.113.168.116 TCP 66 41334 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0
14 5.384768073 192.168.0.100 140.113.168.116 HTTP 595 GET /Project2/pro2-3.html HTTP/1.1
15 5.386646570 140.113.168.116 192.168.0.100 TCP 66 80 → 41334 [ACK] Seq=1 Ack=530 Win=30080 Len=0
16 5.388560936 140.113.168.116 192.168.0.100 HTTP 865 HTTP/1.1 200 OK (text/html)
17 5.388571597 192.168.0.100 140.113.168.116 TCP 66 41334 → 80 [ACK] Seq=530 Ack=800 Win=30080 Len=0
18 18.296381465 140.113.168.116 192.168.0.100 TCP 66 80 → 41334 [FIN, ACK] Seq=800 Ack=530 Win=0 Len=0
19 18.349277087 192.168.0.100 140.113.168.116 TCP 66 41334 → 80 [ACK] Seq=530 Ack=801 Win=30080 Len=0
172 55.496303039 192.168.0.100 140.113.168.116 TCP 66 [TCP Keep-Alive] 41334 → 80 [ACK] Seq=530 Ack=801 Win=0 Len=0
173 55.498051136 140.113.168.116 192.168.0.100 TCP 66 [TCP Keep-Alive ACK] 80 → 41334 [ACK] Seq=530 Ack=801 Win=0 Len=0
329 100.553721211 192.168.0.100 140.113.168.116 TCP 66 [TCP Keep-Alive] 41334 → 80 [ACK] Seq=530 Ack=801 Win=0 Len=0
329 100.553797522 140.113.168.116 192.168.0.100 TCP 66 80 → 41334 [RST] Seq=801 Win=0 Len=0
5 2.836794154 192.168.0.100 140.113.168.116 TCP 74 41418 → 80 [SYN] Seq=0 Win=29200 Len=0
8 2.838253676 140.113.168.116 192.168.0.100 TCP 74 80 → 41418 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
9 2.838304994 192.168.0.100 140.113.168.116 TCP 66 41418 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0
10 2.838607510 192.168.0.100 140.113.168.116 HTTP 595 GET /Project2/pro2-2.html HTTP/1.1
11 2.841035629 140.113.168.116 192.168.0.100 TCP 66 80 → 41418 [ACK] Seq=1 Ack=530 Win=30080 Len=0
12 2.844105620 140.113.168.116 192.168.0.100 HTTP 2268 HTTP/1.1 200 OK (text/html)
13 2.844113958 192.168.0.100 140.113.168.116 TCP 66 41418 → 80 [ACK] Seq=530 Ack=2203 Win=30080 Len=0
18 7.751396712 140.113.168.116 192.168.0.100 TCP 66 80 → 41418 [FIN, ACK] Seq=2203 Ack=530 Win=0 Len=0
19 7.794552535 192.168.0.100 140.113.168.116 TCP 66 41418 → 80 [ACK] Seq=530 Ack=2204 Win=30080 Len=0
```

可以看見本地端的 IP 為 192.168.0.100 伺服器端為 140.113.168.116

(2) 這題感到相當奇怪，未見 HTTP Segment，將所有 filter 拿掉也沒看見，即便重新開啟 wireshark 加上 sudo 權限或是改用 chrome 都沒有看到 segment，推測可能是放在 HTTP Response 夾帶過來了，在這裡浪費了半小時以致於寫不完 QQ

```
7 0.453632933 192.168.0.100 140.113.168.116 TCP 76 41892 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1252059979 TSecr=0 WS=128
8 0.455403335 140.113.168.116 192.168.0.100 TCP 76 80 → 41892 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=416710153 TSecr=1252059979 WS=128
9 0.455427711 192.168.0.100 140.113.168.116 TCP 68 41892 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=1252059980 TSecr=416710153
10 0.455558771 192.168.0.100 140.113.168.116 HTTP 480 GET /Project2/pro2.1.html HTTP/1.1
11 0.457571663 140.113.168.116 192.168.0.100 TCP 68 80 → 41892 [ACK] Seq=1 Ack=413 Win=30080 Len=0 TSval=416710153 TSecr=1252059980
12 0.459309192 140.113.168.116 192.168.0.100 HTTP 867 HTTP/1.1 200 OK (text/html)
13 0.459319395 192.168.0.100 140.113.168.116 TCP 68 41892 → 80 [ACK] Seq=413 Ack=800 Win=30848 Len=0 TSval=1252059981 TSecr=416710154
14 0.472323170 192.168.0.100 140.113.168.116 HTTP 452 GET /favicon.ico HTTP/1.1
15 0.475044310 140.113.168.116 192.168.0.100 HTTP 577 HTTP/1.1 404 Not Found (text/html)
16 0.515180329 192.168.0.100 140.113.168.116 TCP 68 41892 → 80 [ACK] Seq=797 Ack=1309 Win=32512 Len=0 TSval=1252059995 TSecr=416710158
10 2.902560979 140.113.168.116 192.168.0.100 TCP 76 80 → 41808 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=416669960 TSecr=1252019788
13 2.984710954 140.113.168.116 192.168.0.100 TCP 68 80 → 41808 [ACK] Seq=1 Ack=530 Win=30080 Len=0 TSval=416669961 TSecr=1252019788
14 2.987641264 140.113.168.116 192.168.0.100 HTTP 2270 HTTP/1.1 200 OK (text/html)
21 7.894375512 140.113.168.116 192.168.0.100 TCP 68 80 → 41808 [FIN, ACK] Seq=2203 Ack=530 Win=30080 Len=0 TSval=416671188 TSecr=1252019789
```

(3) 當 GET 成功時 回應 200、phrase OK

2.

(1)分別連上如圖中的網站

```
62 6.080951147 192.168.0.100 140.113.168.116 HTTP 529 GET /Project2/pro2-3.html HTTP/1.1
65 6.091926741 192.168.0.100 140.113.168.116 TCP 66 42284 → 80 [ACK] Seq=464 Ack=924 Win=30848 Len=0 TSval=1252401128 TSecr=417051304
66 6.095903279 192.168.0.100 140.113.199.40 TCP 74 54292 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=430244715 TSecr=0 WS=128
73 6.708929915 192.168.0.100 140.113.199.40 TCP 54 54292 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0
74 6.701028812 192.168.0.100 140.113.199.40 HTTP 502 GET /templates/nctunewweb/images/NCTU%20loho_y.png HTTP/1.1
83 6.704124720 192.168.0.100 140.113.199.40 TCP 54 54292 → 80 [ACK] Seq=449 Ack=184 Win=30336 Len=0
89 6.699636849 192.168.0.100 140.113.235.47 TCP 74 58006 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2099442948 TSecr=0 WS=128
79 6.702167814 192.168.0.100 140.113.235.47 TCP 66 58006 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=2099442949 TSecr=3199447531
80 6.702310593 192.168.0.100 140.113.235.47 HTTP 496 GET /~wwchung/pic_logo.png HTTP/1.1
```

而帶有圖片的 imgsrc 為

/templates/nctunewweb/images/NCTU%20loho_y.png

和/~wwchung/pic_logo.png

(2)

No.74 送出 HTTP Request 要求第一張圖片,但 No.73 已經開始去建第一張圖片的 TCP 連線了,且 No.79 開始建下一張圖片的 TCP 連線,但 No.80 送出 HTTP Request,自上述現象可判斷應是平行下載。至於為何兩次的 TCP 連線建立有延遲,大致可判斷是因為 CPU 執行時間所導致 (clock freq vs time)。

62	6.686951147	192.168.0.100	140.113.168.116	HTTP	529 GET /Project2/pro2_3.html HTTP/1.1
65	6.691836741	192.168.0.100	140.113.168.116	TCP	66 42284 → 80 [ACK] Seq=464 Ack=824 Win=30848 Len=0 TSval=1252401128 TSecr=417051304
66	6.699593279	192.168.0.100	140.113.199.40	TCP	74 54292 → 80 [SYN] Seq=0 Win=29312 Len=0 MSS=1460 SACK_PERM=1 TSval=430244715 TSecr=0 WS=128
73	6.700929915	192.168.0.100	140.113.199.40	TCP	54 54292 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0
74	6.701028812	192.168.0.100	140.113.199.40	HTTP	502 GET /templates/nctunewweb/images/NCTU%20logo_v.png HTTP/1.1
83	6.704124720	192.168.0.100	140.113.199.40	TCP	54 54292 → 80 [ACK] Seq=449 Ack=184 Win=30336 Len=0
69	6.699636849	192.168.0.100	140.113.235.47	TCP	74 58006 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2099442948 TSecr=0 WS=128
79	6.702167814	192.168.0.100	140.113.235.47	TCP	66 58006 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=2099442949 TSecr=3199447531
80	6.702310593	192.168.0.100	140.113.235.47	HTTP	496 GET /-wchung/pic_logo.png HTTP/1.1

3.

(1)尚未輸入帳號密碼時,因為代表尚未授權登入,因此在 GET 之後回傳了一個 Unauthorized 尚未授權

22	5.310290063	192.168.0.100	140.113.168.116	HTTP	428 GET /Project2/HTTP-wireshark-file5.html HTTP/1.1
23	5.310802540	140.113.168.116	192.168.0.100	TCP	76 80 → 42412 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=417219638 TSecr=1252569462 WS=128
24	5.310820405	192.168.0.100	140.113.168.116	TCP	68 42412 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=1252569462 TSecr=417219638
25	5.312172973	140.113.168.116	192.168.0.100	TCP	68 80 → 42414 [ACK] Seq=1 Ack=361 Win=30869 Len=0 TSval=417219638 TSecr=1252569462
26	5.313587475	140.113.168.116	192.168.0.100	HTTP	804 HTTP/1.1 401 Unauthorized (text/html)

(2)授權之後可以看到在 HTTP GET 信息中找到一個名為 Authorization 的新 filed。

```
GET /Project2/HTTP-wireshark-file5.html HTTP/1.1
Host: nctucs_icn.nctu.me
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:57.0) Gecko/20100101 Firefox/57.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Authorization: Basic aWNuOnJvY2tz
Connection: keep-alive
Upgrade-Insecure-Requests: 1
If-Modified-Since: Sat, 28 Oct 2017 10:02:39 GMT
If-None-Match: "df-55c98816d9e85-gzip"
Cache-Control: max-age=0
```

HTTP基本認證

本條目需要補充更多來源。 (2013年4月20日)

請協助添加多方面可靠來源以改善這篇條目，無法查證的內容可能會因為異議提出而移除。

在HTTP中，基本認證 (Basic access authentication) 是一種用來允許網頁瀏覽器或其他用戶端程式在請求時提供使用者名稱和口令形式的身份憑證的一種登入驗證方式。

在傳送之前是以使用者名稱追加一個冒號然後串接上口令，並將得出的結果字串再用Base64演算法編碼。例如，提供的使用者名稱是Aladdin、口令是open sesame，則拼接後的結果就是Aladdin:open sesame，然後再將其用Base64編碼，得到QWxhZGRpbjpvcGVuIHNlc2FtZQ==。最終將Base64編碼的字串傳送出去，由接收者解碼得到一個由冒號分隔的使用者名稱和口令的字串。

雖然對使用者名稱和口令的Base64演算法編碼結果很難用肉眼識別解碼，但它仍可以極為輕鬆地被電腦所解碼，就像其容易編碼一樣。編碼這一步驟的目的並不是安全與隱私，而是為將使用者名稱和口令中的不相容的字元轉換為均與HTTP協定相容的字元。

HTTP

HTTP版本

HTTP/0.9 · HTTP/1.0 · HTTP/1.1 · HTTP/2

HTTP請求方法

OPTIONS · GET · HEAD · POST · PUT · DELETE · TRACE · CONNECT · PATCH

頭碼位

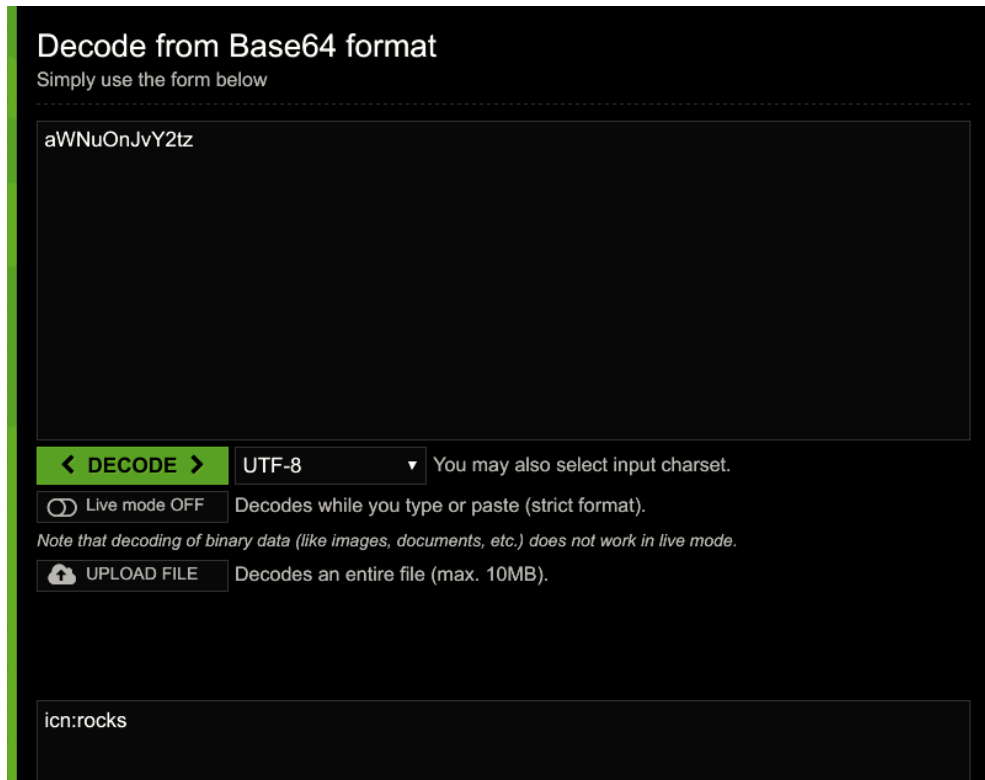
Cookie · ETag · Location · HTTP referer · DNT · X-Forwarded-For

狀態碼

301 Moved Permanently · 302 Found · 303 See Other · 307 Temporary Redirect · 403 Forbidden · 404 Not Found · 451 Unavailable For Legal Reasons

查詢維基百科後發現這是一種加密認證的機制,採用 base64 加密演算法的基本認證 因此回傳以下 header: Authorization(認證) Basic(基本認證) aWNuOnJvY2tz(加密過後的密碼)

4.bonus



The screenshot shows a web application titled "Decode from Base64 format". Below the title, it says "Simply use the form below". There is a large text input area containing the Base64 string "aWNuOnJvY2tz". Below the input area, there is a row of controls: a green button with a left arrow, the word "DECODE", and a right arrow; a dropdown menu currently set to "UTF-8"; and a text label "You may also select input charset.". Below these controls, there are two options: "Live mode OFF" with a toggle switch icon and the text "Decodes while you type or paste (strict format).", and "UPLOAD FILE" with a cloud icon and the text "Decodes an entire file (max. 10MB).". A note below these options states: "Note that decoding of binary data (like images, documents, etc.) does not work in live mode.". At the bottom of the interface, there is a text output area displaying the decoded result "icn:rocks".

可以發現附在 HTTP header 加密的流程是

帳戶名稱：對應密碼 --- (Base-64 encryption algorithm) --> 暗文

以夾帶在 HTTP 中

5.心得

由於大三才修習計網概，相對能有寫作業的時間較少，前幾周瘋狂的微處理機、編譯器設計、OS、機器學習等等的作業忙不過來，只好將計網概的作業放到最後寫，沒想到 wireshark 時而收到封包時而又空白（乙太網路流量正常），令我相當納悶，重開好多次才好。

雖然遲交了一小時，卻也不因為為了趕死線隨邊唬弄作業，在慢慢研究期中的知識後逐漸對網際網路的運作有了更深一層的理解，例如 handshaking ACK NAK 協定，理解 TCP 為何能穩定傳輸以及最重要的資訊安全加密機制，均是課本上無法學到的實做內容。