JARINGAN KOMPUTER

Disusun Oleh :

Nama : Febrina Setianingsih

NIM : 09011181419021

Dosen Pembimbing :

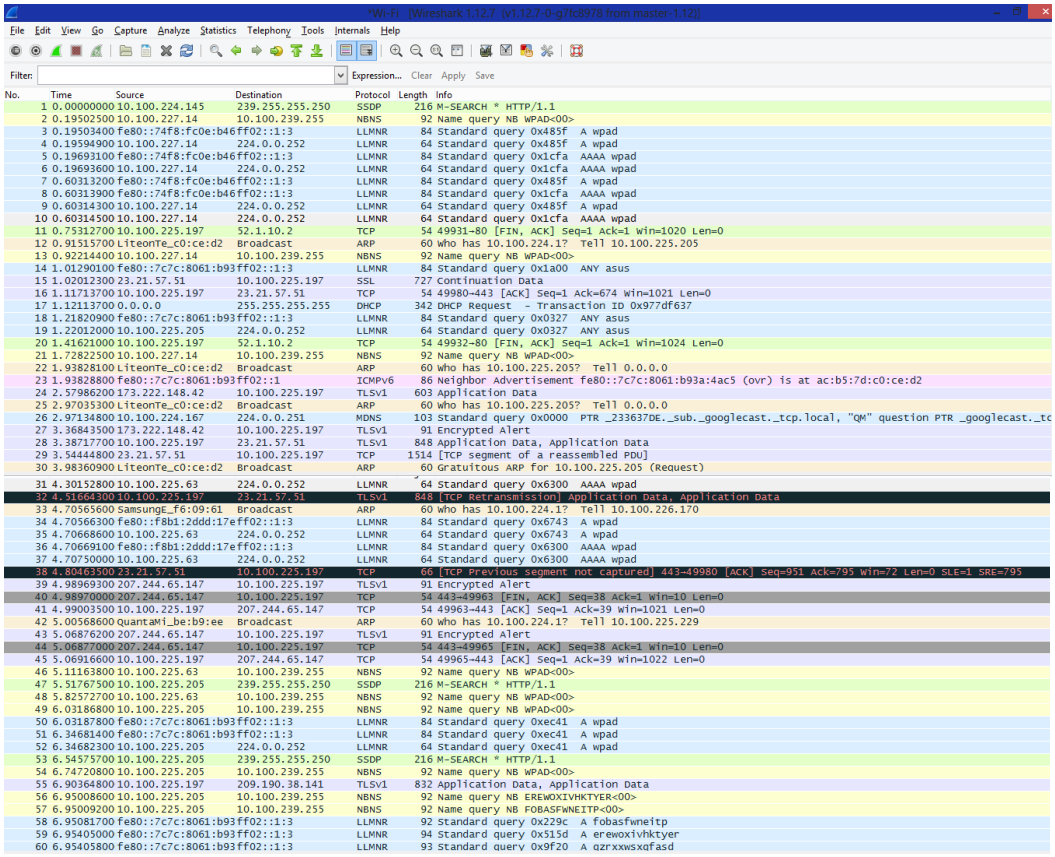
Dr. Deris Stiawan, M.T., Ph.D.

SISTEM KOMPUTER

FAKULTAS ILMU KOMPUTER

UNIVERSITAS SRIWIJAYA

Analisa Frame dari Wireshark



Gambar diatas merupakan hasil capture traffic packet data dengan menggunakan wireshark. Semua jenis paket informasi dalam berbagai format protokol dapat ditangkap dan dianalisa kinerja jaringannya. Wireshark akan menampilkan semua informasi tentang paket yang keluar dan masuk dalam interface yang telah dipilih. Pertama kita harus mengaktifkan wireshark setelah itu kita membuka browser kemudian mengetikkan alamat yang kita inginkan, misalnya www.cisco.com maka wireshark akan mengcapture packet secara lengkap. Pada panel ini berisi nomor paket, waktu saat paket di capture, sumber dan tujuan dari paket, protokol yang digunakan dan panjangnya, serta informasi dari hasil filter paket.

ANALISA PROTOCOL HTTP (GET)

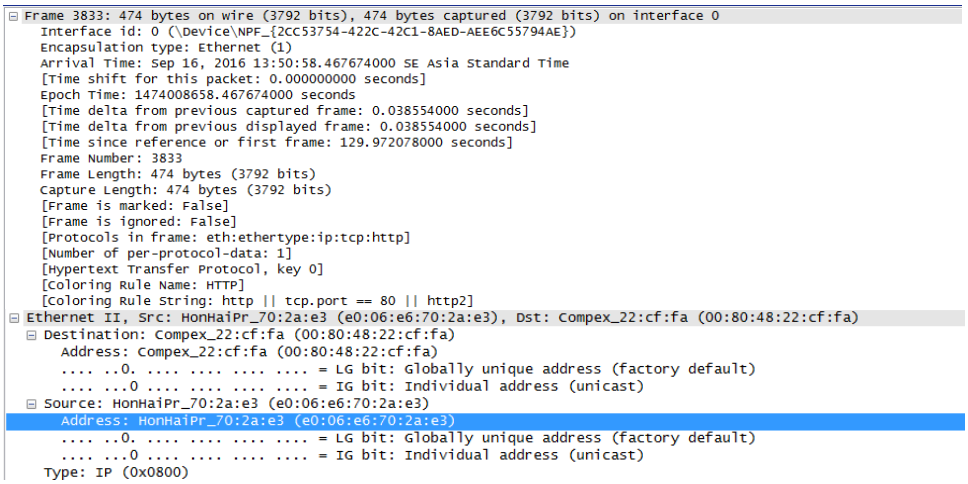
3833 129.972078 10.100.225.197 23.15.100.240 HTTP 474 GET /utag/cisco/cdc/prod/utag.2311.js?utv=ut4.42.201609061332 HTTP/1.1

Sebuah permintaan GET mengambil data dari web server dengan menentukan parameter di bagian URL dari permintaan.

1. MAC Address

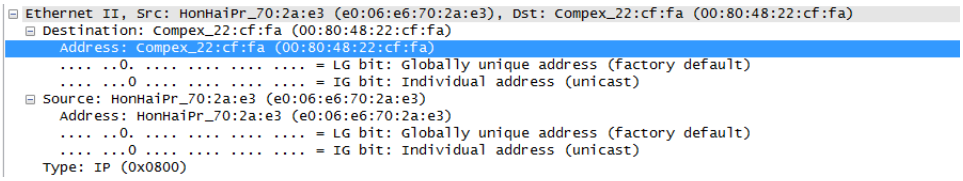
Source MAC Address

= HonHaiPr_70:2a:e3 (e0:06:e6:70:2a:e3)



Destination MAC Address

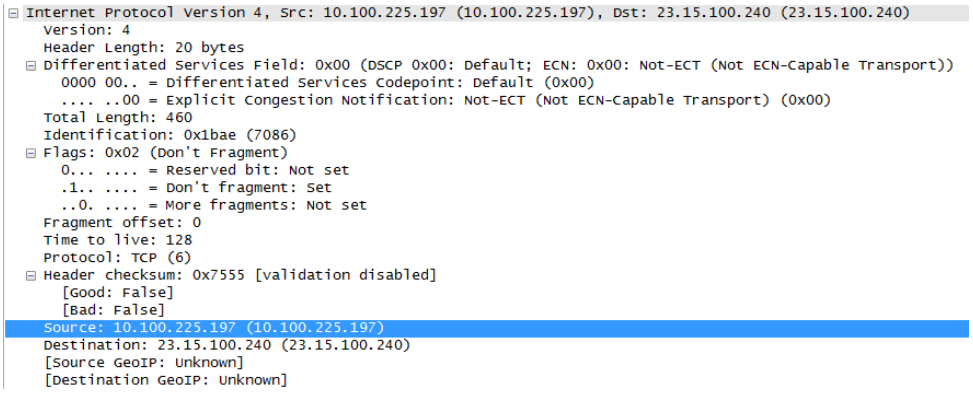
= Compex_22:cf:fa (00:80:48:22:cf:fa)



2. IP Address

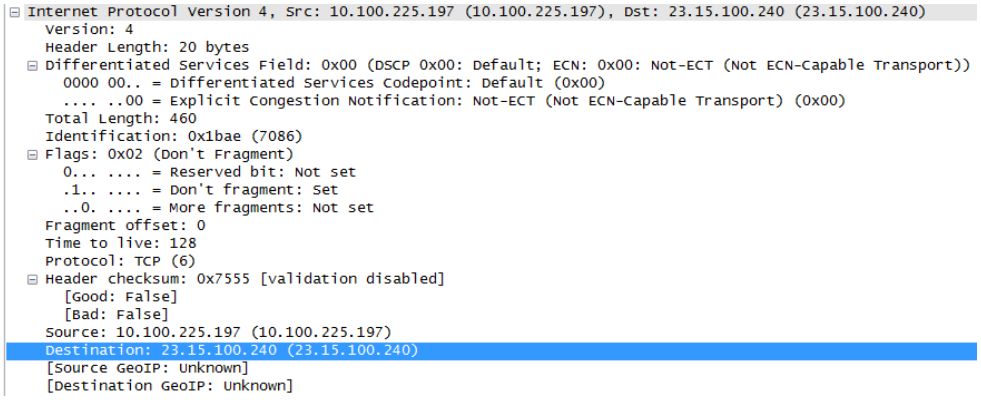
Source IP Address

= 10:100:225:197



Destination IP Address

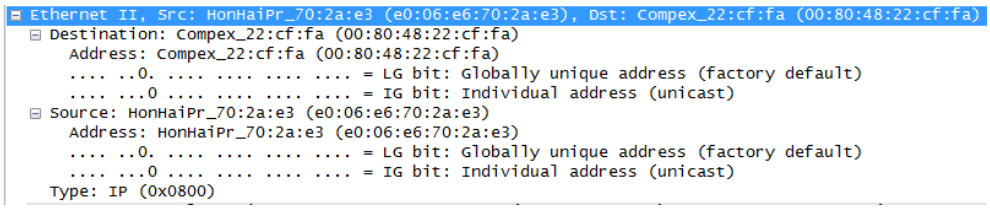
= 23.15.100.240



3. Ukuran L2 dan L3

L2 (Layer 2)

Pada layer 2 terdapat alamat fisik. Alamat fisik adalah sistem pengalamatan pada jaringan komputer yang dikenal dengan nama Media Access Control Address (MAC address). MAC Address merupakan sistem pengalamatan yang menggunakan metode 48 bit. Alamat ini terletak pada Ethernet II. Ukuran pada layer ini adalah 14 byte. Dapat dilihat pada gambar



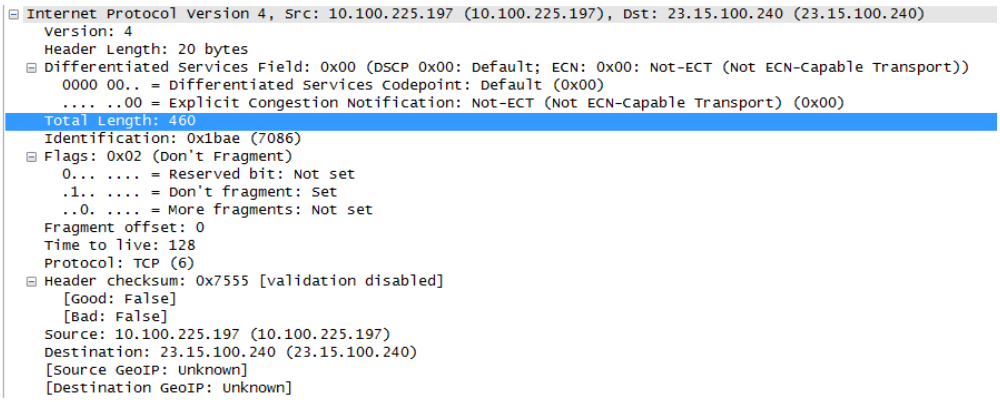
Pada gambar dapat dilihat bahwa pada Ethernet II terdiri dari :

a. DA (Destination MAC Address) yang memiliki panjang 6 byte
b. SA (Source MAC Address) yang memiliki panjang yang sama yaitu 6 byte, dan
c. Type (Type Ethernet) sepanjang 2 byte.

Jadi, pada Ethernet II ini memiliki ukuran 14 byte.

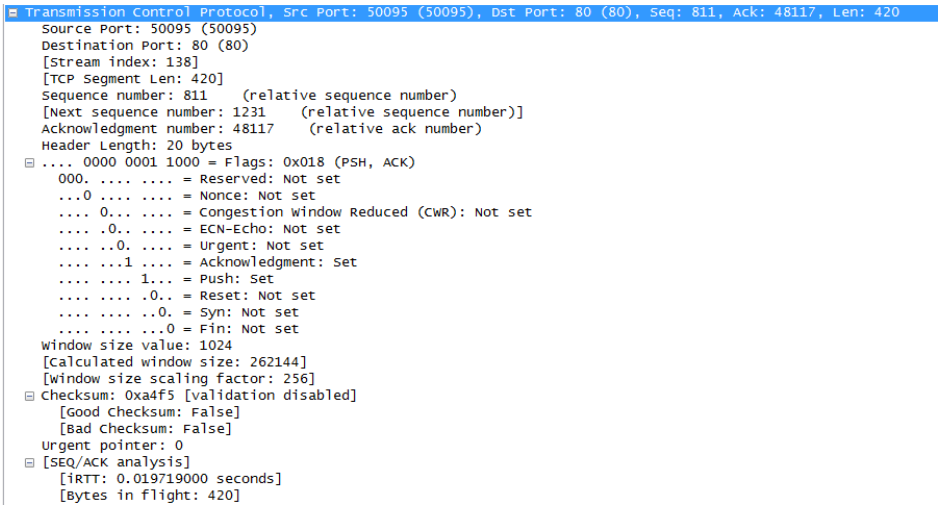
L3 (Layer 3)

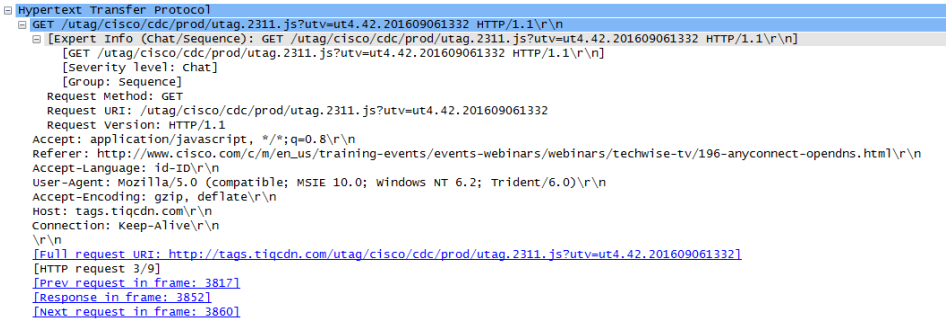
Pada Layer 3 atau Network layer terdapat alamat logika. Alamat logika menggunakan sebuah aturan atau metode yang dikenal dengan nama Internet Protocol address (IP address). Saat ini IP address terdiri atas dua versi yaitu IPv4 dan IPv6. Yang terdapat pada wireshark ini yaitu IPv4. Ukuran layer ini dapat dilihat pada total length-nya yaitu sebesar 460 byte.



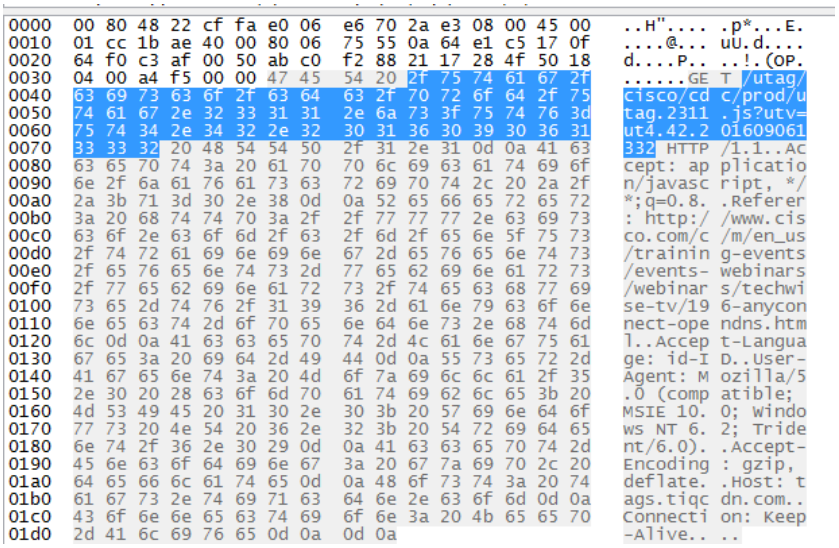
4. Protocol

Protocol yang digunakan pada proses ini adalah protokol HTTP (Hyper Text Transfer Protocol).



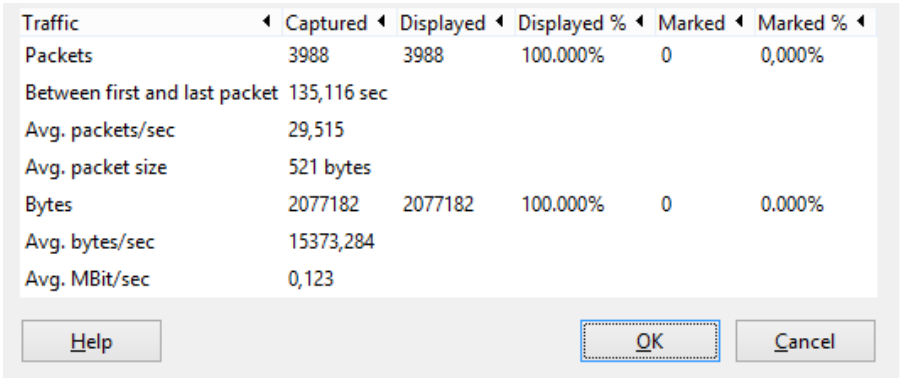


Gambar dibawah ini merupakan bytes dari paket yang berisi data yang diterima atau dikirim dalam bentuk hexadesimal dan disamping kanan merupakan terjemahan dari data hexadesimal tersebut.

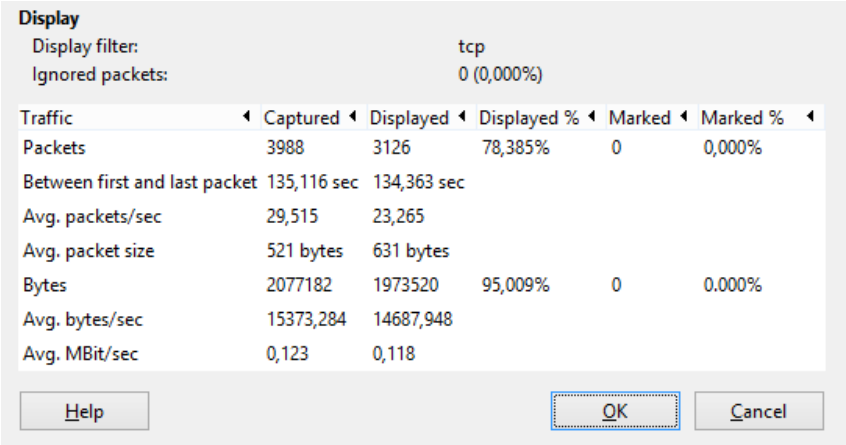


5. Ukuran paket data

Disini saya membandingkan 2 protokol. Ukuran paket data yang ada pada wireshark dengan protokol HTTP yaitu sebanyak 3988 paket. Dan ukuran paket yang menghubungkan antara Host pengirim dan Host penerima ialah sebanyak 3988 paket atau sebanyak 100,000% dari total paket yang ada.



Sedangkan ukuran paket data yang ada pada wireshark dengan protokol TCP yaitu sebanyak 3988 paket. Tetapi ukuran paket yang menghubungkan antara Host pengirim dan Host penerima yaitu sebanyak 3126 paket atau sebanyak 78,385% dari total paket yang ada.



Display

Display filter:	tcp
Ignored packets:	0 (0,000%)

Traffic	Captured	Displayed	Displayed %	Marked	Marked %
Packets	3988	3126	78,385%	0	0,000%
Between first and last packet	135,116 sec	134,363 sec			
Avg. packets/sec	29,515	23,265			
Avg. packet size	521 bytes	631 bytes			
Bytes	2077182	1973520	95,009%	0	0.000%
Avg. bytes/sec	15373,284	14687,948			
Avg. MBit/sec	0,123	0,118			

Help OK Cancel

PROTOCOL HTTP (POST)

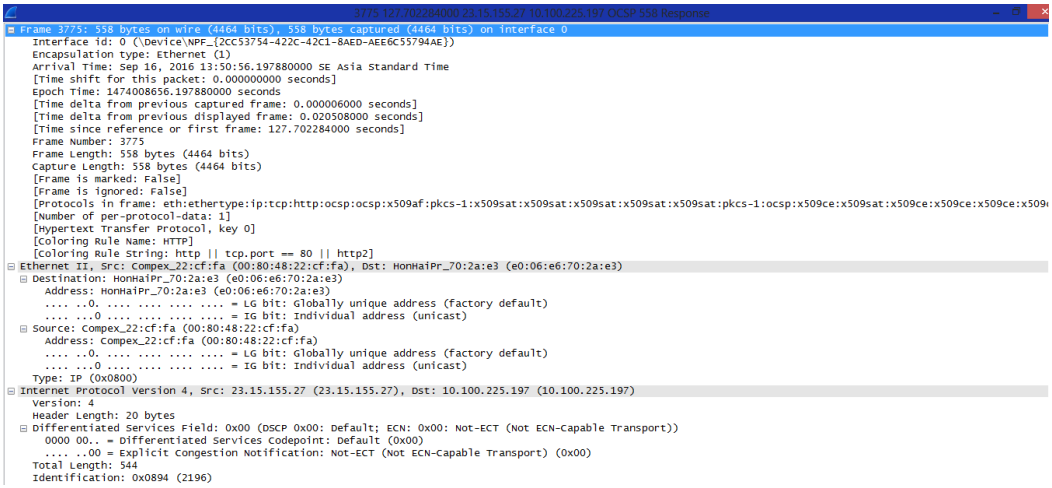
```
321 15.000595000 10.100.225.197 119.81.145.162 HTTP 266 POST /urlinfo/v3/_MD/008123425208F6A8000001575A2992C863646561616231320000015728B6CE51/14740072601 HTTP/1.1
Frame 321: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits) on interface 0
    Interface id: 0 (\Device\NPF_{2CC53754-422C-42C1-8AED-AEE6C55794AE})
    Encapsulation type: Ethernet (1)
    Arrival Time: Sep 16, 2016 13:49:03.496191000 SE Asia Standard Time
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1474008543.496191000 seconds
    [Time delta from previous captured frame: 0.000600000 seconds]
    [Time delta from previous displayed frame: 2.842018000 seconds]
    [Time since reference or first frame: 15.000595000 seconds]
    Frame Number: 321
    Frame Length: 266 bytes (2128 bits)
    Capture Length: 266 bytes (2128 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:tcp:http:data]
    [Number of per-protocol-data: 1]
    [Hypertext Transfer Protocol, key 0]
    [Coloring Rule Name: HTTP]
    [Coloring Rule String: http || tcp.port == 80 || http2]
Ethernet II, Src: HonHaiPr_70:2a:e3 (e0:06:e6:70:2a:e3), Dst: Compex_22:cf:fa (00:80:48:22:cf:fa)
    Destination: Compex_22:cf:fa (00:80:48:22:cf:fa)
        Address: Compex_22:cf:fa (00:80:48:22:cf:fa)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Source: HonHaiPr_70:2a:e3 (e0:06:e6:70:2a:e3)
        Address: HonHaiPr_70:2a:e3 (e0:06:e6:70:2a:e3)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 10.100.225.197 (10.100.225.197), Dst: 119.81.145.162 (119.81.145.162)
    Version: 4
    Header Length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
    Total Length: 252
    Identification: 0x7f39 (32569)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 128
    Protocol: TCP (6)
    Header checksum: 0x85a5 [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 10.100.225.197 (10.100.225.197)
    Destination: 119.81.145.162 (119.81.145.162)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 49970 (49970), Dst Port: 80 (80), Seq: 255, Ack: 1, Len: 212
    Source Port: 49970 (49970)
    Destination Port: 80 (80)
    [Stream index: 21]
    [TCP Segment Len: 212]
    Sequence number: 255    (relative sequence number)
    [Next sequence number: 467    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Header Length: 20 bytes
```

```
    Cache-Control: no-cache\r\n
    Connection: Keep-Alive\r\n
    Pragma: no-cache\r\n
    User-Agent: avast! WRC IE plugin\r\n
    Content-Length: 212\r\n
        [Content length: 212]
    Host: ui.ff.avast.com\r\n
    \r\n
    [Full request URI: http://ui.ff.avast.com/urlinfo/v3/_MD/008123425208F6A8000001575A2992C863646561616231320000015728B6CE51/14740072601]
    [HTTP request 1/1]
    [Response in frame: 325]
Data (212 bytes)
    Data: 0d3301e5721a5ab2f71043ad8629da5de84158898826e741...
    [Length: 212]
```

Protokol HTTP merupakan sebuah protokol yang meminta/menjawab antara klien dan server. Klien HTTP memulai permintaan dengan membuat hubungan ke port tertentu di sebuah server webhosting tertentu (port 80) melalui entity body dalam HTTP Request dengan permintaan post.

PROTOCOL OCSP (RESPONSE)

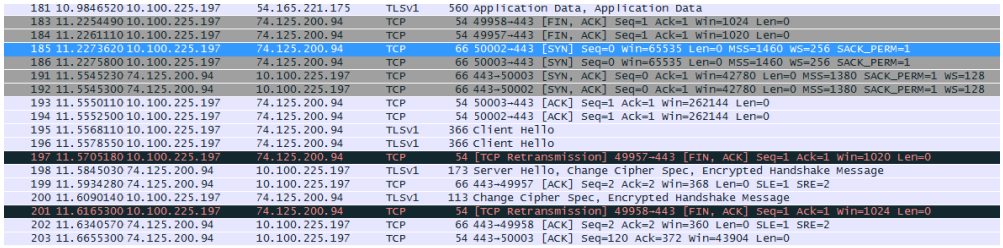
Protokol OCSP (online certificate status protocol). Pada protokol ini



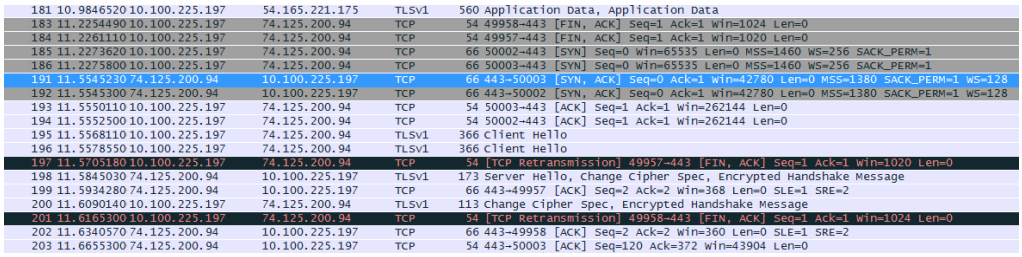
ANALISA PROTOCOL TCP

Pada wireshark ini banyak sekali paket data yang terbentuk dengan protocol yang berbeda-beda. Pada protocol TCP ini kita dapat melihat beberapa protocol jenis TCP dengan info yang berbeda-beda dan saya akan menunjukkan MAC address dan IP address saya. Dari berbagai protocol yang dapat kita lihat, ada tahap pengkoneksian sebuah komputer yang mengirim data (Host pengirim) ke destination (Host Penerima).

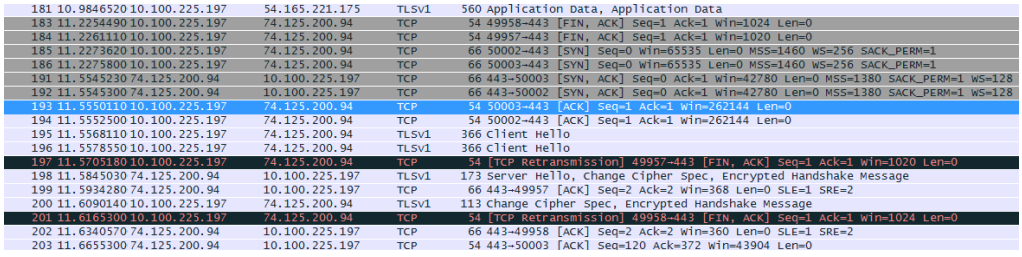
1. Pada awalnya Host pengirim (yaitu komputer saya) mengirim paket SYN ke Host penerima, hal ini dapat dilihat pada gambar dibawah ini (yang telah diberi tanda garis biru).



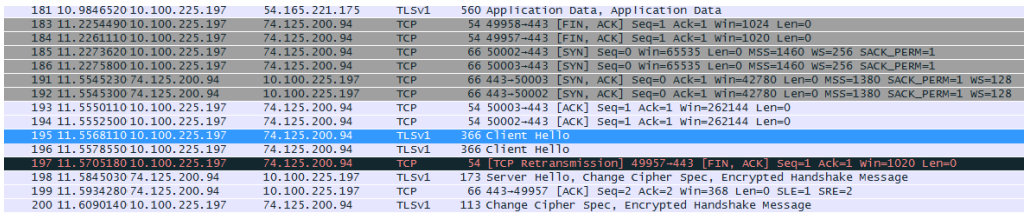
2. Setelah Host penerima menerima paket tersebut kemudian Host penerima mengirim paket SYN serta meng-ACK paket SYN tersebut, hal ini dapat dilihat pada gambar dibawah ini (yang telah diberi tanda garis biru).



3. Setelah Host pengirim menerima paket ini, maka Host Pengirim akan meng-ACK paket tersebut ke Host penerima (dapat dilihat pada gambar a). Setelah itu maka koneksi pun terbuka, dapat dilihat pada gambar b

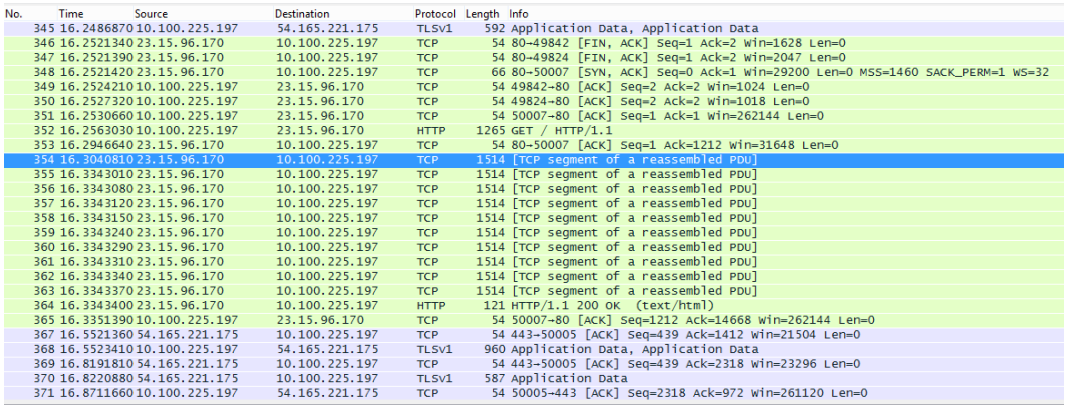


Gambar a. Pada saat Host Pengirim mengirimkan ACK ke Host penerima



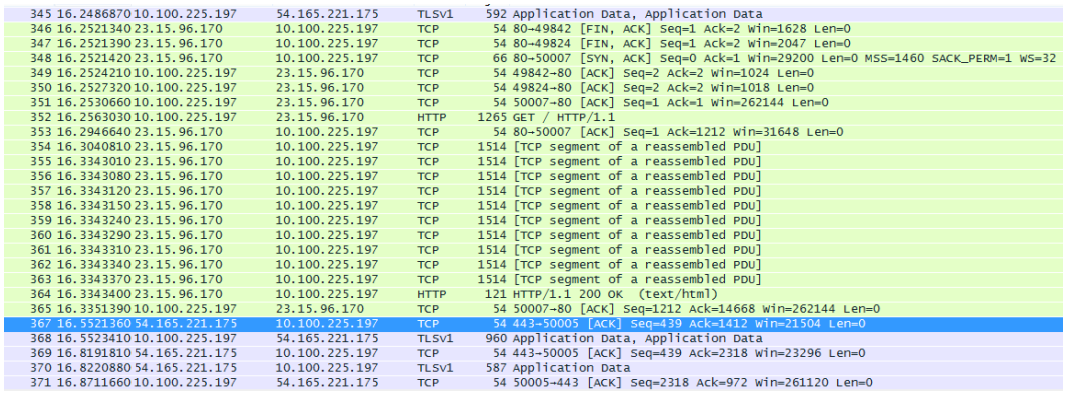
Gambar b. Pada saat koneksi terbuka dengan client dan server saling menyapa

4. Setiap kali sebuah paket tiba di Host Penerima, maka Host penerima akan mengeceknya dan mengirim ACK yang menandakan paket telah diterima. Proses ini berlangsung berulang-ulang sampai semua paket dikirim dan tiba dengan baik.



No.	Time	Source	Destination	Protocol	Length	Info
345	16.2486870	10.100.225.197	54.165.221.175	TLSv1	592	Application Data, Application Data
346	16.2521340	23.15.96.170	10.100.225.197	TCP	54	80→49842 [FIN, ACK] Seq=1 Ack=2 Win=1628 Len=0
347	16.2521390	23.15.96.170	10.100.225.197	TCP	54	80→49824 [FIN, ACK] Seq=1 Ack=2 Win=2047 Len=0
348	16.2521420	23.15.96.170	10.100.225.197	TCP	66	80→50007 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
349	16.2524210	10.100.225.197	23.15.96.170	TCP	54	49842→80 [ACK] Seq=2 Ack=2 Win=1024 Len=0
350	16.2527320	10.100.225.197	23.15.96.170	TCP	54	49824→80 [ACK] Seq=2 Ack=2 Win=1018 Len=0
351	16.2530660	10.100.225.197	23.15.96.170	TCP	54	50007→80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
352	16.2563030	10.100.225.197	23.15.96.170	HTTP	1265	GET / HTTP/1.1
353	16.2946640	23.15.96.170	10.100.225.197	TCP	54	80→50007 [ACK] Seq=1 Ack=1212 Win=31648 Len=0
354	16.3040810	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
355	16.3343010	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
356	16.3343080	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
357	16.3343120	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
358	16.3343150	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
359	16.3343240	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
360	16.3343290	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
361	16.3343310	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
362	16.3343340	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
363	16.3343370	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
364	16.3343400	23.15.96.170	10.100.225.197	HTTP	121	HTTP/1.1 200 OK (text/html)
365	16.3351390	10.100.225.197	23.15.96.170	TCP	54	50007→80 [ACK] Seq=1212 Ack=14668 Win=262144 Len=0
367	16.5521360	54.165.221.175	10.100.225.197	TCP	54	443→50005 [ACK] Seq=439 Ack=1412 Win=21504 Len=0
368	16.5523410	10.100.225.197	54.165.221.175	TLSv1	960	Application Data, Application Data
369	16.8191810	54.165.221.175	10.100.225.197	TCP	54	443→50005 [ACK] Seq=439 Ack=2318 Win=23296 Len=0
370	16.8220880	54.165.221.175	10.100.225.197	TLSv1	587	Application Data
371	16.8711660	10.100.225.197	54.165.221.175	TCP	54	50005→443 [ACK] Seq=2318 Ack=972 Win=261120 Len=0

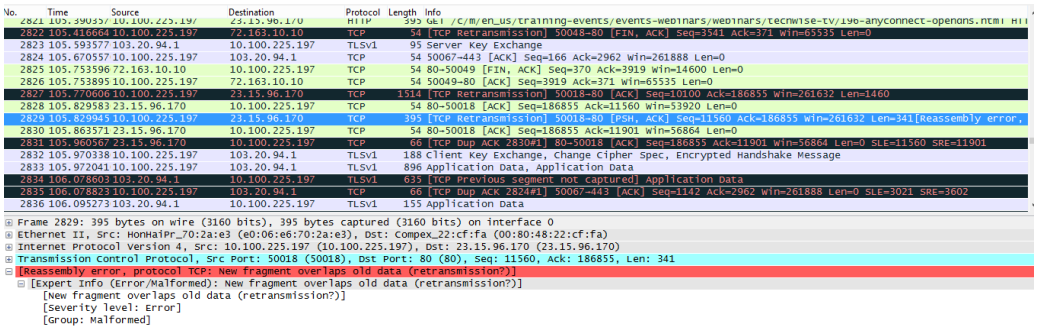
Gambar c. Pada saat Host pengirim mengirim data ke Host penerima



345	16.2486870	10.100.225.197	54.165.221.175	TLSv1	592	Application Data, Application Data
346	16.2521340	23.15.96.170	10.100.225.197	TCP	54	80→49842 [FIN, ACK] Seq=1 Ack=2 Win=1628 Len=0
347	16.2521390	23.15.96.170	10.100.225.197	TCP	54	80→49824 [FIN, ACK] Seq=1 Ack=2 Win=2047 Len=0
348	16.2521420	23.15.96.170	10.100.225.197	TCP	66	80→50007 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
349	16.2524210	10.100.225.197	23.15.96.170	TCP	54	49842→80 [ACK] Seq=2 Ack=2 Win=1024 Len=0
350	16.2527320	10.100.225.197	23.15.96.170	TCP	54	49824→80 [ACK] Seq=2 Ack=2 Win=1018 Len=0
351	16.2530660	10.100.225.197	23.15.96.170	TCP	54	50007→80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
352	16.2563030	10.100.225.197	23.15.96.170	HTTP	1265	GET / HTTP/1.1
353	16.2946640	23.15.96.170	10.100.225.197	TCP	54	80→50007 [ACK] Seq=1 Ack=1212 Win=31648 Len=0
354	16.3040810	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
355	16.3343010	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
356	16.3343080	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
357	16.3343120	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
358	16.3343150	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
359	16.3343240	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
360	16.3343290	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
361	16.3343310	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
362	16.3343340	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
363	16.3343370	23.15.96.170	10.100.225.197	TCP	1514	[TCP segment of a reassembled PDU]
364	16.3343400	23.15.96.170	10.100.225.197	HTTP	121	HTTP/1.1 200 OK (text/html)
365	16.3351390	10.100.225.197	23.15.96.170	TCP	54	50007→80 [ACK] Seq=1212 Ack=14668 Win=262144 Len=0
367	16.5521360	54.165.221.175	10.100.225.197	TCP	54	443→50005 [ACK] Seq=439 Ack=1412 Win=21504 Len=0
368	16.5523410	10.100.225.197	54.165.221.175	TLSv1	960	Application Data, Application Data
369	16.8191810	54.165.221.175	10.100.225.197	TCP	54	443→50005 [ACK] Seq=439 Ack=2318 Win=23296 Len=0
370	16.8220880	54.165.221.175	10.100.225.197	TLSv1	587	Application Data
371	16.8711660	10.100.225.197	54.165.221.175	TCP	54	50005→443 [ACK] Seq=2318 Ack=972 Win=261120 Len=0

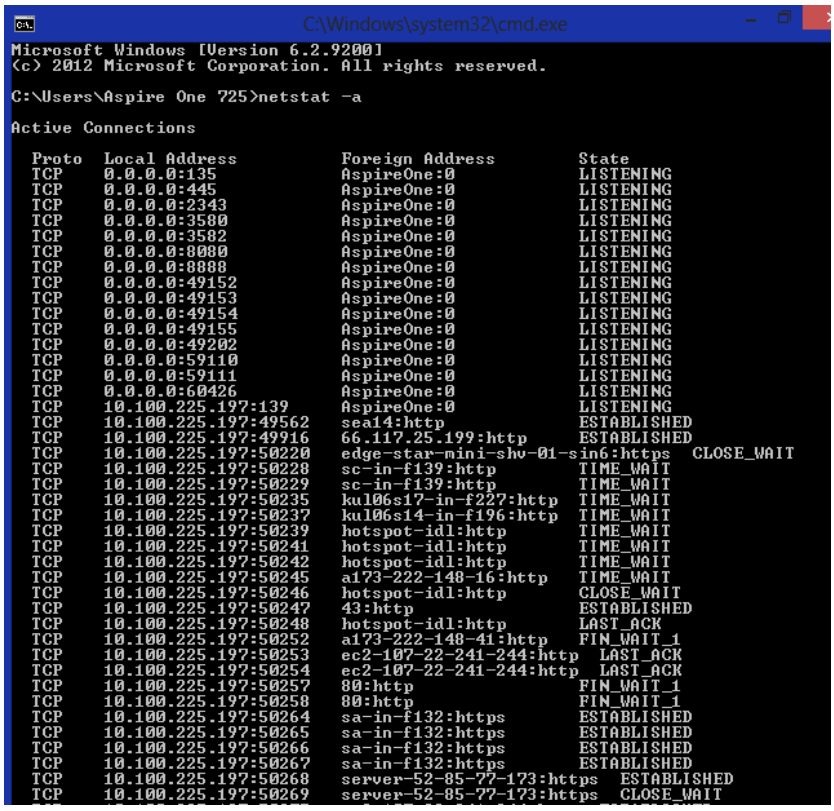
Gambar d. Pada saat Host penerima memberikan pesan dari Host pengirim jika paket telah diterima dengan baik

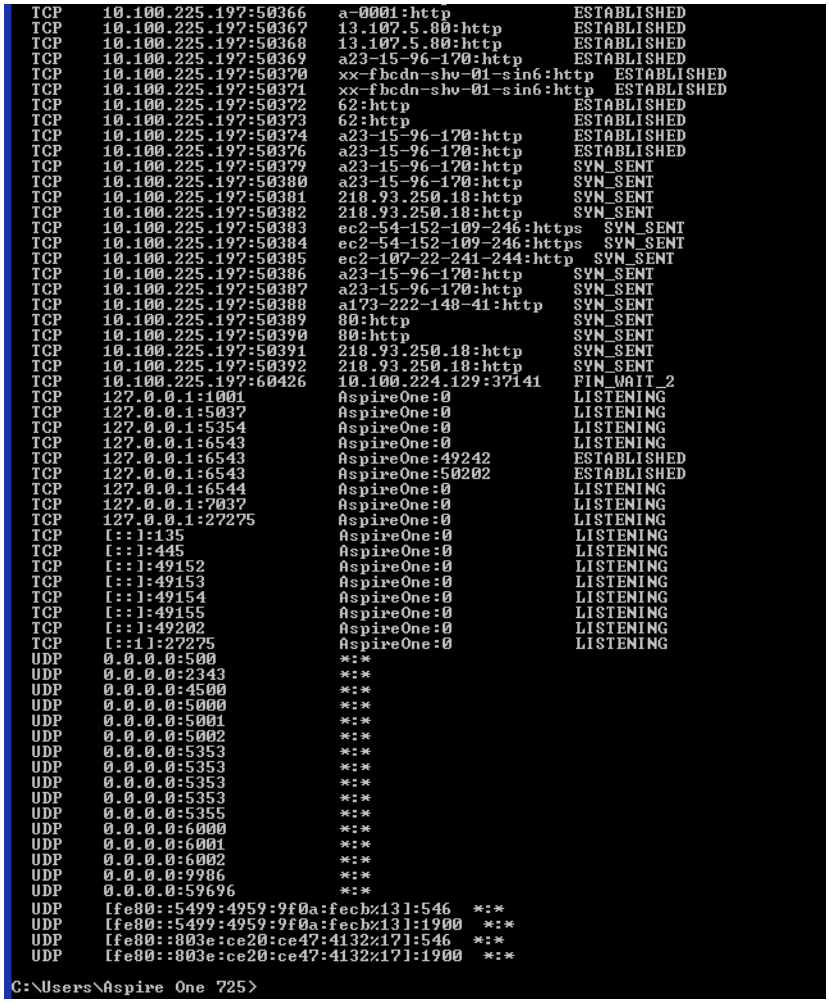
5. Pada percobaan ini saya menemukan bahwa paket dapat hilang atau rusak ditengah jalan, maka Host penerima akan memberitahukan Host pengirim dengan meminta kembali Host pengirim untuk mengirimkan data tersebut. Dapat diliihat pada gambar



Gambar e. Pada saat Host penerima menandakan data dari Host pengirim tidak diterima dengan baik dan meminta pengiriman data kembali.

Analisa Frame dari Command Prompt

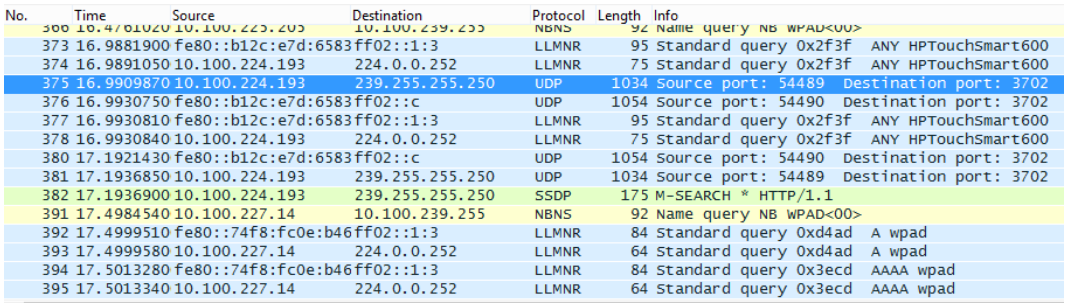




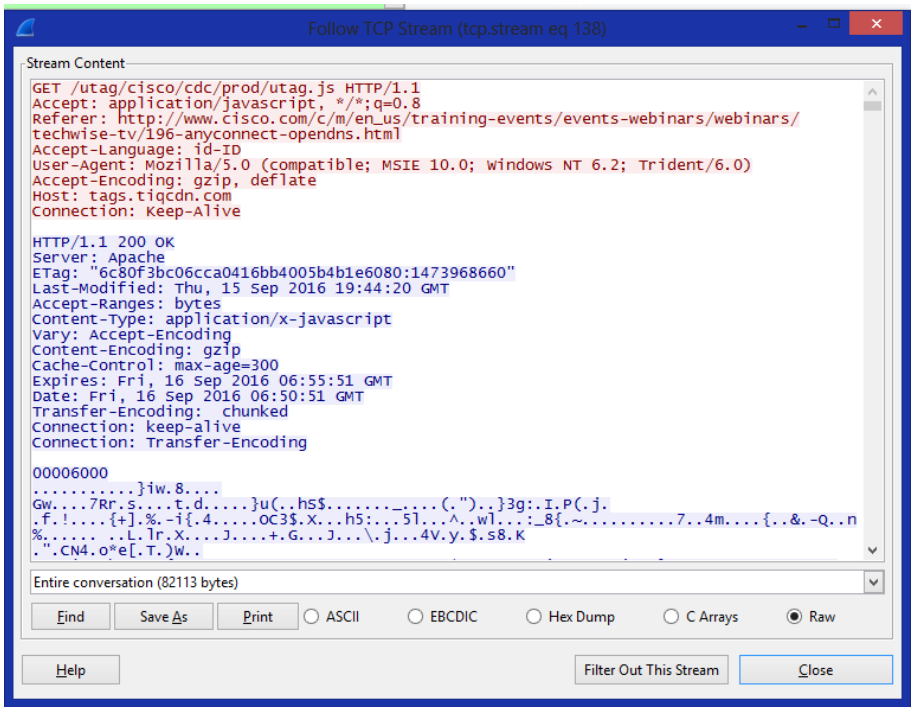
Ketika kita menulis perintah “netstat –a” pada command prompt maka akan muncul tampilan seperti pada gambar diatas. Ada 2 protokol yang muncul pada command prompt ini, yaitu TCP (Transfer Control Protocol) dan UDP (User Data Protocol). Protokol merupakan bahasa yang dimengerti oleh sender to destination. UDP merupakan protokol lapisan transpor TCP/IP yang mendukung komunikasi unreliable tanpa koneksi antara host. Netstat digunakan untuk monitoring traffic jaringan, melacak sumber paket yang didapat, menampilkan statistik protokol dan TCP/IP koneksi jaringan dan informasi yang terkait dengan koneksi port, antara lain alamat IP yang terhubung, status koneksi dan aplikasi yang digunakan.

Komputer berkomunikasi dengan menggunakan protokol TCP dan Local Address (IP address) yang aktif melakukan koneksi dengan Foreign Address. Untuk port-nya dapat dilihat disebelah host komputer. State (status koneksi) yang sering terjadi

seperti LISTENING (siap untuk melakukan koneksi), ESTABLISHED (koneksi terjadi dan siap mengirimkan data), TIME_WAIT (sedang menunggu koneksi), SYS_SENT (mengirimkan paket SYS), FIN_WAIT_1 (bersiap untuk meninggalkan server), TIME_WAIT (siap keluar dari server), CLOSE_WAIT (server mencatat history), FIN_WAIT_2 (server telah mengijinkan client untuk meninggalkan server), dan LAST_ACK (server mengantar client keluar dari server)



No.	Time	Source	Destination	Protocol	Length	Info
373	16.9881900	fe80::b12c:e7d:6583	ff02::1:3	LLMNR	95	Standard query 0x2f3f ANY HPTouchSmart600
374	16.9891050	10.100.224.193	224.0.0.252	LLMNR	75	Standard query 0x2f3f ANY HPTouchSmart600
375	16.9909870	10.100.224.193	239.255.255.250	UDP	1034	Source port: 54489 Destination port: 3702
376	16.9930750	fe80::b12c:e7d:6583	ff02::c	UDP	1054	Source port: 54490 Destination port: 3702
377	16.9930810	fe80::b12c:e7d:6583	ff02::1:3	LLMNR	95	Standard query 0x2f3f ANY HPTouchSmart600
378	16.9930840	10.100.224.193	224.0.0.252	LLMNR	75	Standard query 0x2f3f ANY HPTouchSmart600
380	17.1921430	fe80::b12c:e7d:6583	ff02::c	UDP	1054	Source port: 54490 Destination port: 3702
381	17.1936850	10.100.224.193	239.255.255.250	UDP	1034	Source port: 54489 Destination port: 3702
382	17.1936900	10.100.224.193	239.255.255.250	SSDP	175	M-SEARCH * HTTP/1.1
391	17.4984540	10.100.227.14	10.100.239.255	NBNS	92	Name query NB WPAD<00>
392	17.4999510	fe80::74f8:fc0e:b46	ff02::1:3	LLMNR	84	Standard query 0xd4ad A wpad
393	17.4999580	10.100.227.14	224.0.0.252	LLMNR	64	Standard query 0xd4ad A wpad
394	17.5013280	fe80::74f8:fc0e:b46	ff02::1:3	LLMNR	84	Standard query 0x3ecd AAAA wpad
395	17.5013340	10.100.227.14	224.0.0.252	LLMNR	64	Standard query 0x3ecd AAAA wpad



Port sumber dan port tujuan pada wireshark dapat dilihat pada protokol UDP. Ini merupakan hasil protokol UDP yang telah di capture dari wireshark. Bagian yang berwarna merah merupakan request sent (permintaan dikirim). Dan bagian berwarna biru merupakan response received (respon diterima).

ANALISA

Disini kita dapat melihat perbandingan antara wireshark dan netstat dimana pada wireshark secara terus menerus melakukan pengambilan nilai atau data untuk diamati pada saat proses berlangsung. Sedangkan pada netstat, pengambilan data lebih sedikit dan hanya dilakukan pada saat tertentu secara periodik. Dapat kita lihat juga pada netstat, ada protokol TCP dimana ada reply dari destination sedangkan pada protokol UDP tidak ada reply dari destination tetapi pada protokol ini hemat bandwidth.

KESIMPULAN

- Sebuah komputer yang ingin mengirim data harus mengirim SYN terlebih dahulu kemudian SYN tersebut di ACK dan komputer pengirim meng-ACK maka komunikasi akan terbuka.
- MAC sebuah komputer tidak dapat berubah-ubah sedangkan IP dapat berubah.
- Komunikasi yang dijalankan pada proses ini adalah komunikasi Half-duplex, yaitu komunikasi secara bergantian dimana satu komputer dengan komputer lainnya akan bergantian mengirim data dan menerima balasan, tidak dapat secara langsung.
- Sebuah komputer yang mengirim data, maka komputer penerima akan memberikan balasan dengan cara meng-ACK ke komputer pengirim, tetapi jika data tersebut hilang atau rusak, maka komputer penerima akan meng-ACK komputer pengirim untuk meminta data dikirim ulang (Retransmission Data).