# **TASK 5 JARINGAN KOMPUTER**



Disusun oleh : Nama : Ilham Kholfihim M NIM : 09011281419043

# JURUSAN SISTEM KOMPUTER FAKULTAS ILMU KOMPUTER UNIVERSITAS SRIWIJAYA 2016

## ANALISIS PERBANDINGAN CAPTURING NETWORK

## TRAFFIC MENGGUNAKAN WIRESHARK DAN NETSTAT

Capturing atau Menangkap trafik jaringan sangat berguna bagi administrator untuk menyelesaikan masalah-masalah yang muncul di jaringan, termasuk masalah keamanan. Hal ini berdasarkan fakta bahwa sejumlah penyerangan dimulai dengan penggunaan penyadap untuk melihat trafik jaringan dengan harapan melihat data-data penting yang ditransmisikan oleh user. Karena itu mempelajari kapabilititas dan keterbatasan software-software capturing packet menjadi bagian yang penting dari kemampuan networking.

Pada kegiatan ini saya akan membandingkan trafik jaringan yang dilalui oleh http request dengan menggunakan Wireshark dan Netstat (Network Statistic) pada command line terminal.

HTTP Requests merupakan Permintaan-permintaan dari source ke destination berisikan informasi tentang macam-macam data yang user inginkan.

Siklus dari HTTP Request umumnya terlihat seperti ini:

- 1. Seorang pengguna mengunjungi URL dari sebuah situs web.
- 2. Hal ini menciptakan permintaan yang diarahkan ke web server melalui internet (jaringan DNS itu, router dan switch) melalui HTTP (Hypertext Transfer Protocol).
- 3. Web server menerima permintaan HTTP dan merespon pengguna dengan halaman web (atau isi) yang diminta.

Setiap kali kita klik link dan mengunjungi halaman web, di balik layar kita membuat suatu permintaan, dan menerima respon dari web server. Perhatikan bahwa permintaan HTTP dapat dilakukan melalui berbagai macam jaringan.

Salah satu item informasi yang dienkapsulasi pada permintaan HTTP adalah sebuah method. Hal ini yang memberitahu server macam-macam permintaan yang dibuat, Ada tiga protokol yang biasa digunakan : GET, POST, dan PUT.

- 1. GET adalah method HTTP paling sederhana dan digunakan sebagian besar untuk meminta resource tertentu dari server, dapat berupa halaman web, file gambar grafis, atau sebuah dokumen, dan lain-lain.
- 2. POST, Jenis permintaan ini didesain seperti browser dapat membuat permintaan kompleks dari server. Mereka didesain sehingga user, melalui browser, dapat mengirim banyak data ke server. Form kompleks secara umum dicapai dengan menggunakan permintaan POST, sebagaimana form sederhana yang memelukan proses upload file ke server.

Satu perbedaan yang nyata antara method GET dan POST terletak pada cara mengirimkan data ke server. Seperti yang dinyatakan sebelumnya, GET hanya

menambahkan data ke URL yang akan mengirim. POST, di sisi lain, mengenkapsulasi atau menyembunyikan data di dalam body pesan (message body) yang dikirim. Ketika server menerima permintaan dan menentukan bahwa itu merupakan sebuah permintaan POST, dapat dilihat dari body pesan data tersebut.



- 3. PUT, Berfungsi untuk meng-upload representasi dari sumber tertentu
- 4. HTTP Response, HTTP merespon dari server yang berisi headers dan body pesan, seperti yang permintaan HTTP lakukan. Mereka menggunakan kumpulan header yang berbeda, meskipun demikian disini kita tidak perlu terlalu dalam membahasnya secara detail. Cukup dengan mengatakan bahwa headers berisi informasi tentang protokol HTTP yang digunakan pada server, sebagaimana tipe dari isi yang dienkapsulasi ke dalam body pesan. Nilai dari tipe isi adalah MIME-type. Ini akan memberitahu browser jika pesan berisi HTML, gambar, atau tipe lainnya.

## **1. WIRESHARK**

Wireshark, atau dulunya dikenal sebagai Ethereal, adalah salah satu tool yang sangat powerfull sebagai senjata para analis keamanan jaringan. Sebagai analyzer packet jaringan, Wireshark dapat digunakan sebagai peer di dalam suatu jaringan dan mengamati trafik secara detail dalam berbagai level, mulai dari header packet hingga bit yang menyusan suatu paket. Karena wireshark menggunakan GUI, banyak pengguna memperoleh kemudahan grafis dalam menggunakan informasi yang terkandung di dalamnya.

# 2. NETSTAT

Netstat (network statistics) merupakan program berbasis teks yang berfungsi untuk memantau koneksi jaringan pada suatu komputer, baik itu jaringan lokal (LAN) maupun jaringan internet.

Berikut ini keterangan dari output netstat :

- 1. **Proto**. Kolom proto menunjukan jenis protokol yang dipakai bisa TCP atau UDP.
- 2. Local Address. Kolom ini menjelaskan alamat dan nomor port yang ada di komputer anda yang mana saat itu sedang aktif melakukan koneksi.
- 3. Foreign Address. Kolom ini menunjukan koneksi yang dituju oleh local address beserta nomor portnya. Contoh diatas saya sedang menghubungi server google melalui http (port 80) yang artinya saya sedang browsing google.
- 4. **State**. Kolom ini menunjukan status dari koneksi yang sedang terjadi. **ESTABLISED** artinya sudah terhubung dengan komputer lain dan siap mengirimkan data.

State yang biasa terjadi :

- LISTENING -> siap untuk melakukan koneksi.
- SYN\_SENT -> mengirimkan paket SYN
- **SYN\_RECEIVED** -> menerima paket SYN
- ESTABLISHED -> koneksi terjadi dan siap mengirimkan data.
- TIME\_WAIT -> sedang menunggu koneksi

Perlu diperhatikan jika muncul state SYN\_SENT dalam jumlah yang banyak dan terus menerus, efeknya koneksi internet anda menjadi sangat lambat.

# ANALISA

Capturing data menggunakan wireshark dapat dilihat pada gambar di bawah ini

∕ ⊘		$\odot$ $\odot$ $\otimes$				
File Edit	le Edit View Go Capture Analyze Statistics Telephony Tools Internals Help					
• •	● 📕 🖻 🗎 ● ○   ♀ < > > ≂ ⊻ 🗏 🖥 🛙 其 🗹 🖉 ♦ 캁 🕸					
Filter:	: p.request.method == GET or http.request.method == POST 💌 Expression Clear Apply Save					
No.	Time Source Destination Protocol Length Info					
3957 4944	157         73.84465501(10.100.228.25         216.58.199.206         QCSP         495 Request           444         83.64965152(10.100.228.25         117.18.237.29         QCSP         497 Request					
5055	555         84.72218385         100.228.25         216.58.199.206         OCSP         495         Request           32         84.41144776(10.100.228.25)         74.125.68.154         HTTP         1002         GET         /ativeriew?avi=BpRIQ0K7gV82UJsKSogPSoLS4	DgAAAAAQATgByAEJwAIC4AIA4AQBoAYW&cid=CAAS				
8982 9040	NB2         100,7546561(10.100.228.25         119.81.42.42         HTTP         691         GET         /(ol-9-&ajax_seek=1474335549&seek_max_tim           N40         101.0678125:10.100.228.25         119.81.42.42         HTTP         622         GET         /images/unsave.jpg         HTTP/1.1           N40         101.0678125:10.100.228.25         119.81.42.42         HTTP         622         GET         /images/unsave.jpg         HTTP/1.1	e=1474342448 HTTP/1.1				
9041 9116 9117	HI 101.0722304110.100.228.25 119.81.42.42 HTP 625 GET /nmageS/nsTw.png HTP/1.1 116 101.3279822 10.100.228.25 119.81.42.42 HTTP 655 GET /posts/1024d69a041404beaad8efc6aaa364aa_t 17 10 12944010 10 100 .229 25 119 81.42.16 HTTP 655 GET /nosts/1024d69a041404beaad8efc6aaa364aa_t	.jpg HTTP/1.1				
9797	197 104.1121336:10.100.228.25 119.81.42.42 HTTP 694 GET / HTTP/101 196 104.2156713:10.100.228.25 119.81.42.42 HTTP 694 GET / HTTP/1.1					
+ e	ame 3957; 495 bytes on wire (3960 bits), 495 bytes captured (3960 bits) on interface o					
Ether	ernet II, Src: HonHaiPr_5a:99:e5 (c0:18:85:5a:99:e5), Dst: Compexin_22:cf:fa (00:80:48:22:cf:fa) ternet Bertrel Version 4. Src: 10:100-230-25. Det: 216:58-100-205	I				
<ul> <li>Trans</li> </ul>	refinet Frotocol Version 4, 51c. 10:100-1223:23, 51c. 10:30-139-120 ansmission Control Protocol, Src Port: 43224 (43224), Dst Port: 80 (80), Seq: 1289, Ack: 2239, Len: 429					
► POS	POST /ocsp HTTP/1.1/r/n					
Use	nost: clenisi.gogie.com/in User-Agent: Mozilla/5.0 (XII: Linux x86 64: rv:48.0) Gecko/20100101 Firefox/48.0\r\n					
Aco	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n					
Acc	Accept-Language: en-US,en;q=0.5\r\n	· · · · · · · · · · · · · · · · · · ·				
0000 0	00 80 48 22 cf fa c0 18 85 5a 99 e5 08 00 45 00					
0010 0	0] el 1f 84 40 00 40 06 8b 0c 0a 64 e4 19 d8 3a@.@d:					
0020 C	c/ ce as 08 00 50 51 49 59 21 c0 a1 27 74 80 18P.1 11/					
0040 1	18 b2 50 4f 53 54 20 2f 6f 63 73 70 20 48 54 54POST / OCSP HTT					
0060 6	55 66 74 73 31 26 76 6 6 67 66 65 66 76 66 6 67 66 65 26 63 66 6 entsl.go ogle.com					
• 💅	File: "/home/iamkfm/File Ilham KF Packets: 71237 · Displayed: 78 (0.1%) · Load time: 0:05.674	Profile: Default				

Hasil dari capturing dapat dilihat pada kotak berwarna merah, menampilkan bermacam-macam informasi, yaitu :

- 1. **No,** menampilkan informasi nomor paket ke-n yang diperoleh pada saat melakukan interkoneksi.
- 2. Time, menampilkan informasi waktu mengakses paket dimulai dari start capturing.
- 3. Source, menampilkan informasi IP dari sumber data atau pengakses.
- 4. Destination, menampilkan informasi IP dari tujuan data tersebut.
- 5. **Protokol**, menampilkan informasi Protokol yang digunakan pada saat mengakses data tersebut.
- 6. **Info,** menampilkan informasi aktivitas yang dilakukan oleh data seperti request, send, post, get, dan lain sebagainya.

Pada posisi pointer yang berada didalam kotak berwarna kuning terlihat aktivitas **request** yang sedang dilakukan oleh alamat IP **10.100.228.25** yang merupakan alamat IP yang saya gunakan yang merupakan IP Static Jaringan Hotspot di Fakultas Ilmu Komputer Universitas Sriwijaya yang meminta paket data ke alamat tujuan yaitu suatu alamat website <a href="http://lcak.com">http://lcak.com</a> dengan alamat IP **74.125.68.154** 

Dari proses capturing dapat dilihat suatu kondisi request dimana alamat website yang saya tuju tidak langsung diakses dari jaringan lokal yang saya gunakan, akan tetapi request dilakukan melalui beberapa rute yang melalui port **ocsp** ke alamat IP **216.58.199.206** lalu selanjutnya ke alamat IP **117.18.237.29** dan proses request terakhir ke alamat IP **216.58.199.286**, alamat-alamat tersebut merupakan alamat ISP dan juga alamat Server dari website yang saya tuju.

Setelah melakukan proses request ke server selanjutnya server mengirimkan kembali paket data ke source yang menyatakan bahwa paket data yang diminta terdapat pada server tersebut.

🥖 🎯			capture terbaru.po	apng [Wireshark 2.0.5]	$\odot$ $\odot$ $\otimes$	
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help						
•	) 🛋 🔳 🔬 🖿 🗎 🛛	$O \mid O \prec \rightarrow \rightarrow =$		8 I I 🖬 💐 🖬 🖕 🏛		
Filter:	ter: prequest.method == GET or http:request.method == POST V Expression Clear Apply Save					
No.	Time Source	Destination	Protocol Length	Info		
3957 4944 5055 5532 8982 9040 9041 9116 9117 9797 9797 9860	73.44455011 10.100.228.25 83.64965152(10.100.228.25 84.722183851 10.100.228.25 84.722183851 10.100.228.25 101.6781251 10.100.228.25 101.6781251 10.100.228.25 101.3279822 10.100.228.25 101.3284210 10.100.228.25 101.3284210 10.100.228.25 104.11213361 10.100.228.25 104.11213361 10.100.228.25	216.58.199.206 117.18.237.29 216.58.199.206 74.125.08.194.206 119.81.42.42 119.81.42.42 119.81.42.42 119.81.42.42 119.81.42.42 119.81.42.42 119.81.42.42 119.81.42.42 119.81.42.42 10.81.42	0CSP         495           0CSP         497           0CSP         495           HTTP         1002           HTTP         601           HTTP         621           HTTP         620           HTTP         620           HTTP         655           HTTP         691           HTTP         655           HTTP         632           HTTP         1016           BC (8016 bits) on         1016	Request Request GET /activeview?avi=Bph100K7qV22U3sK5agPG GET /lol-0-6ajax_seek=14743355496seek_max GET /images/unsave.jpg HTTP/1.1 GET /posts/1024d69a041404beaad6efc6aaa364 GET /posts/1024d69a041404beaad6efc6aaa364 GET /posts/1024d69a041404beaad6efc6aaa364 GET /posts/1024d69a041404beaad6efc6aaa364 GET /posts/1024d69a041404beaad6efc6aaa364 GET /sefiewiewiew2aui=BeTDDW7aV82U12K6cnb0 Ditefrace 0	oLS4DgAAAAAAA rgbyAEJWATC4ATA4AQBOAYWGc1d=CAAS time=1474342448 HTTP/1.1 aa_t.jpg HTTP/1.1 63_t.jpg HTTP/1.1 ol GADDAAAAAAAAATATBUAFJUATCAATAAAABBOAYWGcid=CAAS	
<pre>&gt; ream 5352. 1002 bytes on wire topic 01618; 1002 bytes topicited topic 015 on 1162 bytes 142 bytes 1</pre>						
nosr.pageauz.gouguesynalelain.com(rin User-Agent: Mozilla/5.0 (XII; Linux x86_64; rv:48.0) Gecko/20100101 Firefox/48.0\r\n Accept: #/*\r\n Accept-Language: en-US,en;q=0.5\r\n						
0000 0 0010 0 0020 4 0030 0 0040 9 0050 7 0060 3	$ \begin{smallmatrix} 0 & 80 & 48 & 22 & cf fa & c0 & 18 & 85 & 5a \\ 3 & dc & 2b & cb & 40 & 00 & 40 & 06 & 8d & bc \\ 4 & 9a & 98 & 2e & 00 & 50 & fa & 5c & cc & 6c \\ 6 & ed & 09 & 13 & 00 & 00 & 01 & 01 & 08 & 0a \\ e & b9 & 47 & 45 & 54 & 20 & 2f & 61 & 63 & 74 \\ 7 & 3f & 61 & 76 & 69 & 3d & 42 & 76 & 52 & 49 \\ 8 & 32 & 55 & 4a & 73 & 4b & 53 & 6f & 67 & 50 \\ \end{smallmatrix} $	99 e5 08 00 45 00+" 0a 64 e4 19 4a 7d+.@.@ 35 79 91 b8 01 8 DP 00 2f f3 2e c0 5fFr 69 76 65 76 69 65GET / 51 4f 4b 37 67 56 w?avl=B 53 6f 4c 53 34 44 B2UJsKS	ZE. dJ} \ .l5y a ctivevie p RIQOK7gV o gPSoLS4D		Ĵ	
9 🏹	File: "/home/iamkfm/File Ilham KF	Packets: 71237 · Displayed: 78 (0.1%	) · Load time: 0:05.674		Profile: Default	

Lalu kemudian source meminta kembali paket data yang terdapat pada server dengan cara yang sama melalui beberapa rute seperti yang terlihat pada gambar di atas yang ditandai kotak berwarna hijau, Proses yang dilakukan oleh source untuk meminta data ke destination disini merupakan proses GET yang dilakukan melalui port http pada OSI layer proses ini terdapat pada layer ke-3 Layer Network. Pada capturing yang saya lakukan tidak terdapat proses POST karena saya tidak mengisikan form apapun pada website tersebut.

Dengan Menggunakan Wireshark kita bisa mendapatkan beberapa informasi yang lebih mendetail jika dibandingkan dengan menggunakan netstat pada CLI. Beberapa informasi yang diperoleh dapat dilihat pada gambar dibawah ini.

#### 1. Informasi Frame

```
▼ Frame 10717: 761 bytes on wire (6088 bits), 761 bytes captured (6088 bits) on interface 0
    Interface id: 0 (wlp7s0b1)
    Encapsulation type: Ethernet (1)
    Arrival Time: Sep 20, 2016 10:34:36.640360566 WIB
    [Time shift for this packet: 0.00000000 seconds]
    Epoch Time: 1474342476.640360566 seconds
    [Time delta from previous captured frame: 0.000202685 seconds]
    [Time delta from previous displayed frame: 0.157305632 seconds]
    [Time since reference or first frame: 106.800791806 seconds]
    Frame Number: 10717
    Frame Length: 761 bytes (6088 bits)
    Capture Length: 761 bytes (6088 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:tcp:http]
    [Coloring Rule Name: HTTP]
    [Coloring Rule String: http || tcp.port == 80 || http2]
```

Pada capturing yang saya lakukan dapat diperoleh informasi frame seperti gambar diatas seperti waktu frame dieksekusi, jumlah frame, lebar frame, protokol yang digunakan oleh frame yaitu **ethernet:ip:tcp:http.** Pada OSI Layer aktivitas frame terdapat pada layer

#### ke-2 yaitu Datalink.

#### 2. Informasi Interface Network yang digunakan seperti IP dan mac address

Ethernet II, Src: HonHaiPr\_5a:99:e5 (c0:18:85:5a:99:e5), Dst: CompexIn\_22:cf:fa (00:80:48:22:cf:fa)
 Destination: CompexIn\_22:cf:fa (00:80:48:22:cf:fa)

> Source: HonHaiPr\_5a:99:e5 (c0:18:85:5a:99:e5)

```
Type: IPv4 (0x0800)
```

```
• Internet Protocol Version 4, Src: 10.100.228.25, Dst: 119.81.42.42
0100 .... = Version: 4
```

```
.... 0101 = Header Length: 20 bytes (5)
```

- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT) Total Length: 747 Identification: 0x09a5 (2469)
- Flags: 0x02 (Don't Fragment)
- Fragment offset: 0
- Time to live: 64
- Protocol: TCP (6)
- Header checksum: 0x9e6f [validation disabled] Source: 10.100.228.25 Destination: 119.81.42.42 [Source GeoIP: Unknown]
- [Destination GeoIP: Unknown]

#### 3. Informasi TCP

```
Transmission Control Protocol, Src Port: 52296 (52296), Dst Port: 80 (80), Seq: 3834, Ack: 42792, Len: 695
     Source Port: 52296
     Destination Port: 80
     [Stream index: 57]
     [TCP Segment Len: 695]
                               (relative sequence number)
     Sequence number: 3834
    [Next sequence number: 4529 (relative sequence number)]
Acknowledgment number: 42792 (relative ack number)
    Header Length: 32 bytes
  ▶ Flags: 0x018 (PSH, ACK)
     [Calculated window size: 124800]
     [Window size scaling factor: 128]
  Checksum: 0xbd4e [validation disabled]
    Urgent pointer: 0
   ▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [SEQ/ACK analysis]
```

#### 4. Informasi port yang digunakan

```
* Hypertext Transfer Protocol
* Hypertext Transfer Protocol
* GT / posts/4f60c0d03a8341085a3976255946d3b0_tsmall.jpg HTTP/1.1\r\n
Host: Lcak.com\r\n
Ussr-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:48.0) Gecko/20100101 Firefox/48.0\r\n
Accept: */*\r\n
Accept: encoding: gzip, deflate\r\n
Accept: encoding: gzip, deflate\r\n
Referer: http://lcak.com/r\n
! [truncated[cokie: _______dssID=59b5ba57a550c6c2:T=1464233973:S=ALNI_MZTqUdx8lVaKH00Avi60Q9JBEMkDQ; ___utma=222692187.1966080900.1464233970.1473264814.1474342437.10;
Connection: keep-alive\r\n
If-Modified-Since: Mon, 19 Sep 2016 12:46:57 GMT\r\n
If-Modified-Since: Mon, 19 Sep 2016 12:46:57 GMT\r\n
If-Modified-Since: Mon, 19 Sep 2016 12:46:57 GMT\r\n
If-Lone-Match: *57dfde41-c02*\r\n
Cache-Control: max-age=0\r\n
\r\n
If-II: runcatest URI: http://lcak.com/posts/4f60c0d03a8341805a3976255946d3b0_tsmall.jpg]
[HTTP request 7/8]
[Prov_request in frame: 10923]
```

Pada gambar di atas dapat diperoleh informasi port yang dilalui source untuk sampai ke destination. Dalam hal ini saya mengakses website <u>http://lcak.com</u> dari web browser Mozilla Firefox dan system operasi Linux.

Jika di wireshark menampilkan informasi trafik jaringan secara mendetail lain halnya dengan netstat(network statistic) pada CLI di linux maupun CMD di windows.

Seperti yang telah dijelaskan di awal instruksi netstat menampilkan statistik koneksi jaringan dari suatu komputer. Pada gambar dibawah ini merupakan hasil dari capturing yang telah saya lakukan pada saat terjadinya proses request ke website <u>http://1cak.com</u>

Saya menggunakan website destination yang sama agar dapat dengan mudah memahami perbedaannya menggunakan wireshark dan netstat.

[sk	libur@fuk	amuori :	iamkfm]\$ netstat -a		
Act	ive Inter	net con	nections (servers and	d established)	
Pro	to Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	54	fukamuori:45292	202.58.9.201:https	LAST ACK
tcp	0	0	fukamuori:58258	edge-star-mini-sh:https	ESTABLISHED
tcp	0	1	fukamuori:38600	a23-15-155-27.:www-http	FIN WAIT1
tcp	0	0	fukamuori:38606	a23-15-155-27.:www-http	TIME WAIT
tcp	Ø	0	fukamuori:50930	sin04s09-in-f206.:https	ESTABLISHED
tcp	0	0	fukamuori:37082	202.79.210.118:https	TIME_WAIT
tcp	0	0	fukamuori:48378	104.244.43.145:https	TIME_WAIT
tcp	0	0	fukamuori:45294	202.58.9.201:https	TIME_WAIT
tcp	0	0	fukamuori:47496	sb-in-f95.1e100.n:https	ESTABLISHED
tcp	0	0	fukamuori:36644	<pre>sc-in-f155.1e100.:https</pre>	ESTABLISHED
tcp	0	0	fukamuori:51348	kul06s14-in-f194.:https	ESTABLISHED
tcp	0	0	fukamuori:49600	luna.archlinux.or:https	ESTABLISHED
tcp	0	Ø	fukamuori:44118	<pre>sa-in-f139.1e1:www-http</pre>	TIME_WAIT
tcp	0	Ø	fukamuori:33612	74.125.200.84:https	ESTABLISHED
tcp	0	0	fukamuori:54806	74.125.200.94:https	ESTABLISHED
tcp	0	Ø	fukamuori:54944	74.125.68.157:https	ESTABLISHED
tcp	0	Ø	fukamuori:48382	104.244.43.145:https	TIME_WAIT
tcp	Ø	0	fukamuori:49448	74.125.130.156:https	ESTABLISHED
tcp	0	0	fukamuori:59864	216.58.196.46:https	ESTABLISHED
tcp	Ø	0	fukamuori:42276	104.244.42.136:https	ESTABLISHED
tcp	Ø	0	fukamuori:42348	216.58.199.198:https	ESTABLISHED
tcp	0	0	fukamuori:54852	74.125.68.157:https	ESTABLISHED
tcp	Ø	Ø	fukamuori:38934	172.217.24.228:https	ESTABLISHED
tcp	0	1	fukamuori:38602	a23-15-155-27.:www-http	FIN_WAIT1
tcp	0	0	tukamuori:45300	202.58.9.201:https	TIME_WAIT
tcp	0	0	fukamuori:38604	a23-15-155-27.:www-http	TIME_WAIT
tcp	0	124	fukamuori:55330	216.58.196.166:https	FIN_WAIT1

Dari gambar diatas dapat diperoleh beberapa informasi yaitu proto, local address, foreign address, dan state yang telah dijelaskan di awal. Dari gambar diatas diperoleh informasi sebagai berikut.

1. Proto

Protokol yang digunakan yaitu tcp

2. Local Address

Informasi Local Address yang diperoleh disini berupa nama host yang komputer yang saya gunakan yaitu **fukamuori** beserta nomor port yang digunakan. Biasanya informasi yang ditampilkan berupa alamat IP source karena beberapa pengaturan di kernel linux jadi yang ditampilkan adalah hostname.

3. Foreign Address

Sama seperti penjelasan alamat IP tujuan pada wireshark sebelumnya alamat IP tujuan melalui beberapa rute. Pada instruksi netstat port alamat tujuan yang dalam hal ini http ditampilkan bersamaan dengan IP-nya.

### 4. State

State inilah yang dapat perbedaannya paling mencolok antara penggunaan wireshark dan netstat. Pada wireshark telah dijelaskan sebelumnya kondisi state ini dijelaskan secara mendetail di dalam setiap paket yang di capture. Sedangkan pada netstat kondisi statistik ini hanya diekspresikan dalam suatu keadaan. Seperti capturing web browser yang saya lakukan dengan instruksi netstat pada gambar di bawah ini

tcp	0	0	fukamuori:44118	sa-in-f139.1e1:www-http	TIME_WAIT
tcp	0	0	fukamuori:33612	74.125.200.84:https	ESTABLISHED
tcp	0	0	fukamuori:54806	74.125.200.94:https	ESTABLISHED
tcp	0	0	fukamuori:54944	74.125.68.157:https	ESTABLISHED
tcp	Ø	Ø	fukamuori:48382	104.244.43.145:https	TIME_WAIT
tcp	0	0	fukamuori:49448	74.125.130.156:https	ESTABLISHED
tcp	0	0	fukamuori:59864	216.58.196.46:https	ESTABLISHED
tcp	0	0	fukamuori:42276	104.244.42.136:https	ESTABLISHED
tcp	0	Ø	fukamuori:42348	216.58.199.198:https	ESTABLISHED
tcp	0	0	fukamuori:54852	74.125.68.157:https	ESTABLISHED
tcp	0	0	fukamuori:38934	172.217.24.228:https	ESTABLISHED
tcp	0	1	fukamuori:38602	a23-15-155-27.:www-http	FIN_WAIT1
tcp	0	0	fukamuori:45300	202.58.9.201:https	TIME_WAIT
tcp	0	0	fukamuori:38604	a23-15-155-27.:www-http	TIME_WAIT
tcp	0	124	fukamuori:55330	216.58.196.166:https	FIN_WAIT1

Dapat dilihat proses yang terjadi pada saat saya mengakses website <u>http://1cak.com</u> dengan alamat IP **74.125.200.84** yang melalui port tujuan http dimulai dari proses **TIME\_WAIT** maksudnya adalah source sedang menunggu koneksi dari destination yang dianalogikan seperti diagram dibawah ini.





Setelah proses pada diagram di atas terjadi maka akan tercipta proses ESTABLISHED.

Itulah analisa perbandingan beberapa Informasi yang bisa diperoleh dengan menggunakan wireshark dan netstat masih banyak lagi informasi lainnya yang bisa diperoleh dari wireshark, untuk saat ini karena keterbatasan ilmu yang telah saya miliki, saya hanya bisa menjelaskan sebagian kecil proses 3 way handshake yang terjadi pada dunia jaringan komputer yang dapat dilihat melalui wireshark dan netstat.