ST. ANTONY'S COLLEGE

PERUVANTHANAM

AFFILIATED TO MAHATMA GANDHI UNIVERSITY

BSc CYBER FORENSICS

NAME:…………………………………………………………………………….

CLASS:…………………………………SEMESTER……………………...

SUBJECT:……………………………………………………………………

YEAR:…………………..…..REG.NO:………………………………………...

ST. ANTONY'S COLLEGE

PERUVANTHANAM

AFFILIATED TO MAHATMA GANDHI UNIVERSITY

CERTIFICATE

This is to certify that this Record work done in ....by Mr./Ms.. during the year ....in partial fulfillment of the requirement for the award of the degree of ....of Mahatma Gandhi University, Kottayam, Kerala.

Semester:……………………… Reg. No:…………………………..

Faculty-in Charge Dept. Seal Head of the Department

Submitted for the Practical Examination held on………………………………

Internal Examiner External Examiner

INDEX

EXP NO.	DATE	NAME OF EXPERIMENTS	PAGE NO.	REMARKS

Experiment No:-01 Date: …/... /2024

Windows- Network Commands

Aim:

This experiment is to analyze or configure the TCP/IP Networks Using various useful Windows- Network commands.

Useful Windows- Network Commands

1. IPCONFIG

IPCONFIG: Displays or refresh the TCP/IP configuration. ipconfig /all [/release [adapter]] [/renew [adapter]] /flushdns /displaydns /registerdns. This command, when executed with no options, displays the current IP address, the subnet mask and default gateway (network interfaces of the local machine)

- ipconfig : Displays IP address, mask and default gateway.

- ipconfig /all: Displays all network configuration, including DNS, WINS, DHCP servers, etc..

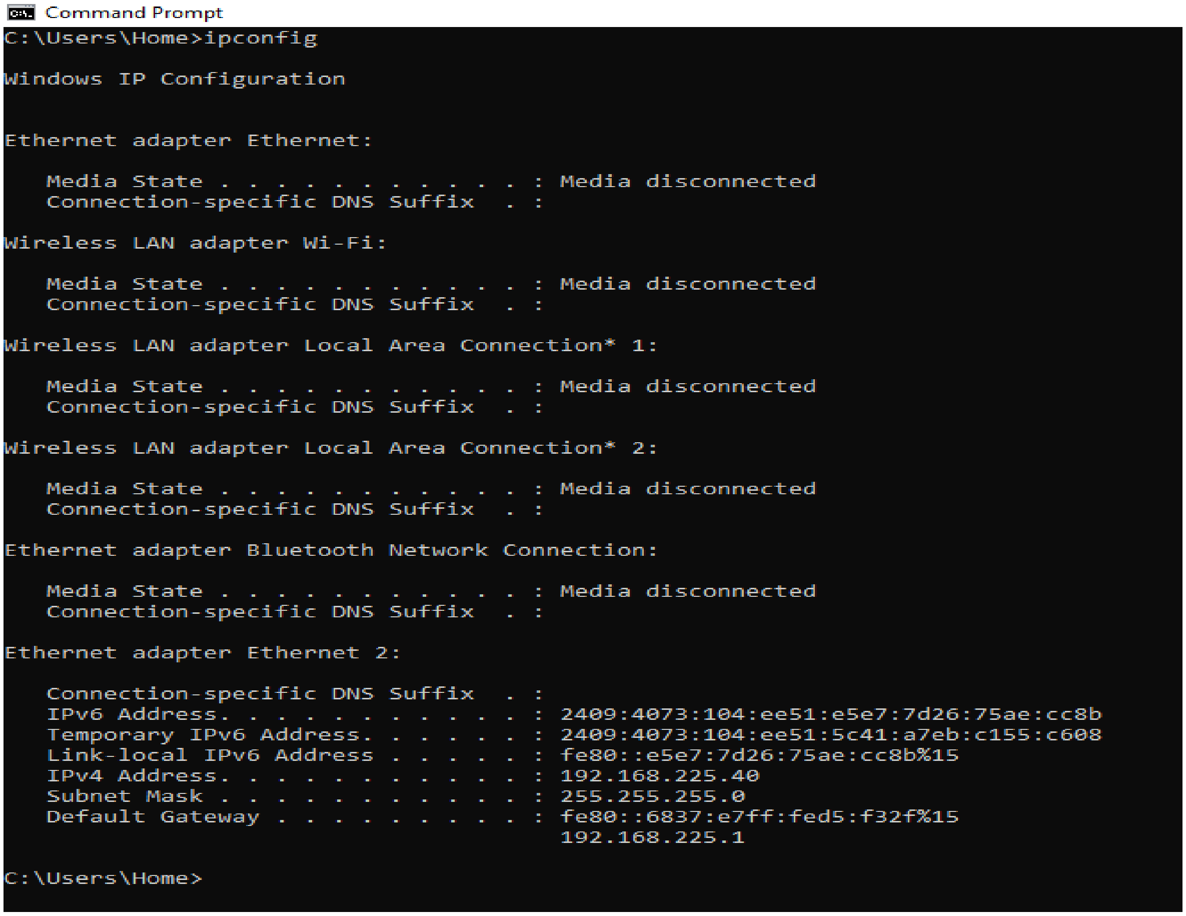
- ipconfig /renew [adapter]:

Renews DHCP configuration for all adapters (if adapter is not specified) or a specific adapter indicated by the [adapter] parameter.

- ipconfig /release [adapter]:

Sends a DHCPRELEASE message to the DHCP server to release the current DHCP configuration and cancel the IP address configuration for all adapters (if adapter is not specified) or a specific adapter indicated by the [adapter] parameter. This parameter disables TCP/IP for network cards configured to automatically obtain an IP address.

1) Screenshot of windows ip configuration output



Command Prompt

```
C:\Users\Home>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2409:4073:104:ee51:e5e7:7d26:75ae:cc8b
   Temporary IPv6 Address. . . . . . : 2409:4073:104:ee51:5c41:a7eb:c155:c608
   Link-local IPv6 Address . . . . . : fe80::e5e7:7d26:75ae:cc8b%15
   IPv4 Address. . . . . . . . . . . : 192.168.225.40
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::6837:e7ff:fed5:f32f%15
                                       192.168.225.1

C:\Users\Home>
```

2. PING

Test a network connectivity - if successful, ping returns the ip address.

Syntax

PING [options] destination_host

A response of "Request timed out" means there was no response within the default time period of 4 seconds. If the latency of the response is more than 4 seconds. Use the -w option to increase the time-out. For example, to allow responses within 10 seconds, use ping -w 10000

The IPv6 options are only available on versions of Windows that support IPv6, e.g. Windows 7 /2008

A successful PING does NOT always return an %errorlevel% of 0. Therefore to reliably detect a successful ping - pipe the output into FIND and look for the text "TTL"

Note that "Reply" in the output of PING does not always indicate a positive response.

Example message: Reply from 192.168.1.254: Destination Net Unreachable.

Ping response times below 10 milliseconds often have low accuracy. A time of 10 milliseconds is roughly equal to a distance of 1860 Miles, travelling a straight line route at the speed of light, (or a round trip of 2 × 930 miles). From this you can see that ping response times will give a very very rough estimate of the distance to a remote host.

Network adminstrators do not appreciate frequent or continual pings to their servers, try not to overdo it!

How to test connectivity with ping:

1) Ping the loopback address to verify that TCP/IP is installed and configured correctly on the local computer.

```
PING 127.0.0.1
```

2) Ping the IP address of the local computer to verify that it was added to the network correctly.

```
PING IP_address_of_local_host
```

3) Ping the IP address of the default gateway to verify that the default gateway is functioning and that you can communicate with a local host on the local network.

```
PING IP_address_of_default_gateway
```

4) Ping the IP address of a remote host to verify that you can communicate through a router.

PING IP_address_of_remote_host

Examples

Check if a host is reachable:

ping example.com

ping 192.168.1.1

Ping a server just once:

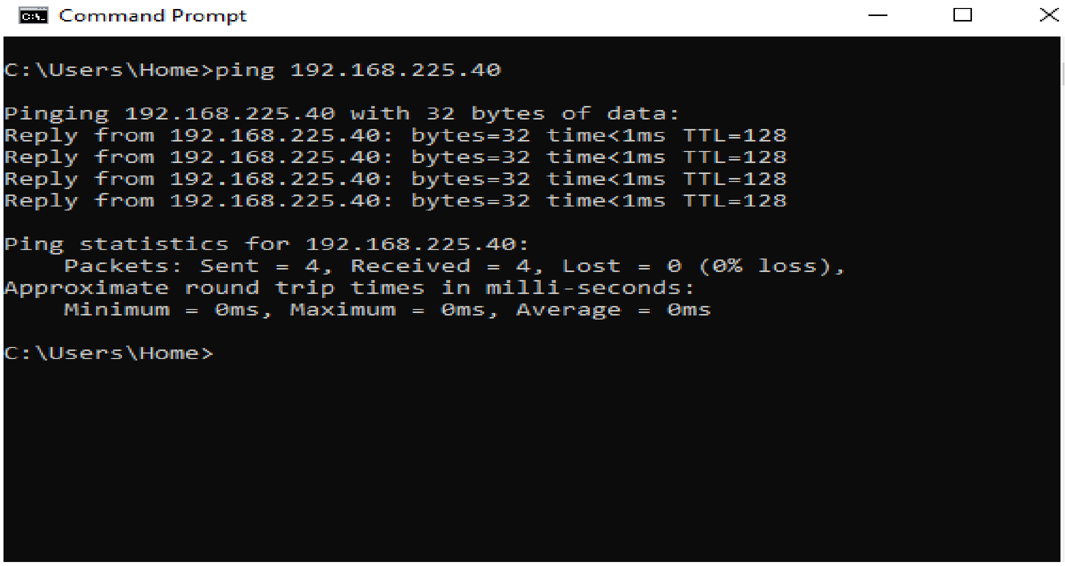
Ping -n 1 example.com

Ping a website 5 times:

PING -n 5 -w 7500 www.microsoft.com

PING is named after the sound that a sonar makes.

1) Screenshot of ping command output



```
C:\Users\Home>ping 192.168.225.40

Pinging 192.168.225.40 with 32 bytes of data:
Reply from 192.168.225.40: bytes=32 time<1ms TTL=128
Reply from 192.168.225.40: bytes=32 time<1ms TTL=128
Reply from 192.168.225.40: bytes=32 time<1ms TTL=128
Reply from 192.168.225.40: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.225.40:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\Home>
```

3. PATHPING

Trace route and provide network latency and packet loss for each router and link in the path. Combines the functionality of PING and TRACERT.

Syntax

```
PATHPING [-n] [-h max_hops] [-g host_list] [-p period]
  [-q num_queries] [-w timeout] [-i IPAddress] [-4 ] [-6 ][TargetName]
```

Key

-g host_list - Loose source route along host-list.

-h max_hops - Maximum number of hops to search for target.

-i address - Use the specified source address.

-n - Do not resolve addresses to hostnames.

-p period - Wait period milliseconds between pings.

-q num_queries - Number of queries per hop.

-w timeout - Wait timeout milliseconds for each reply.

-P - Test for RSVP PATH connectivity.

-R - Test if each hop is RSVP aware.

-T - Test connectivity to each hop with Layer-2 priority tags.

-4 - Force using IPv4.

-6 - Force using IPv6.

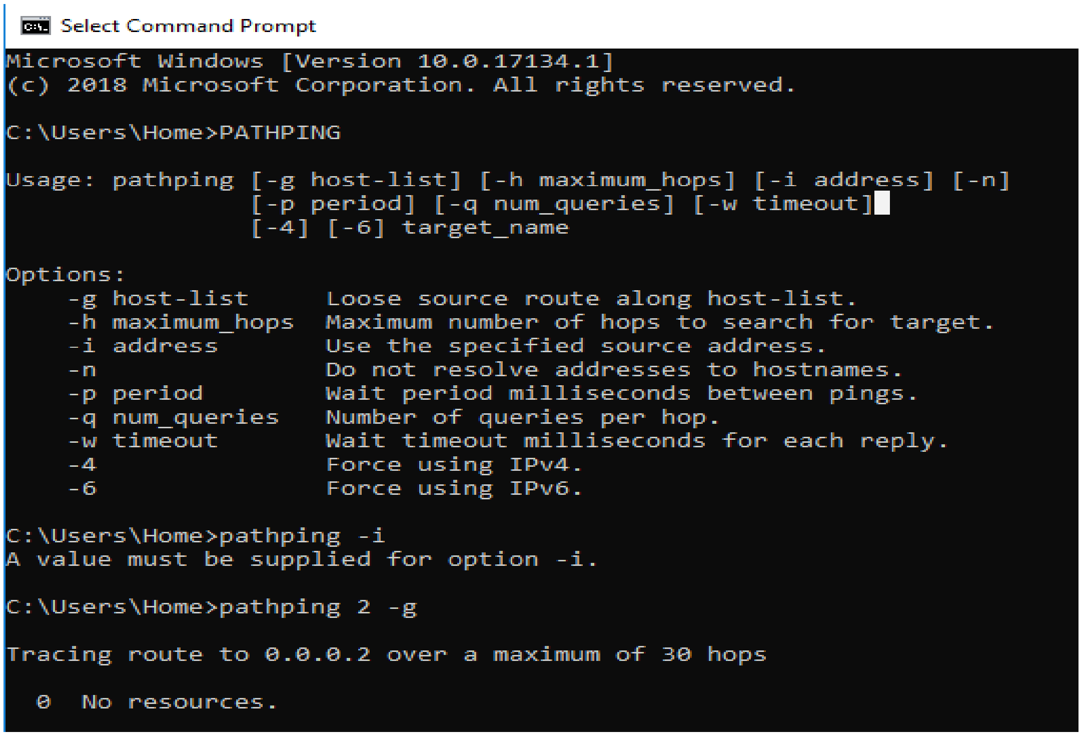
PathPing is invaluable for determining which routers or subnets are having network problems - it displays the degree of packet loss at any given router or link.

Pathping sends multiple Echo Request messages to each router between a source and destination over a period of time and computes aggregate results based on the packets returned from each router. Pathping performs the equivalent of the tracert command by identifying which routers are on the path. To avoid network congestion and to minimize the effect of burst losses, pings should be sent at a sufficiently slow pace (not too frequently.)

When -p is specified, pings are sent individually to each intermediate hop. When -w is specified, multiple pings can be sent in parallel. It's therefore possible to choose a Timeout parameter that is less than the wait Period * Number of hops.

Note: Like tracert PathPing uses Internet Control Message Protocol (ICMP) over TCP/IP. Many firewalls will block ICMP traffic by default. If an attacker is able to forge ICMP redirect packets, he or she can alter the routing tables on the host and possibly subvert the security of the host by causing traffic to flow via a path you didn't intend.

1) Screenshot of pathping command output

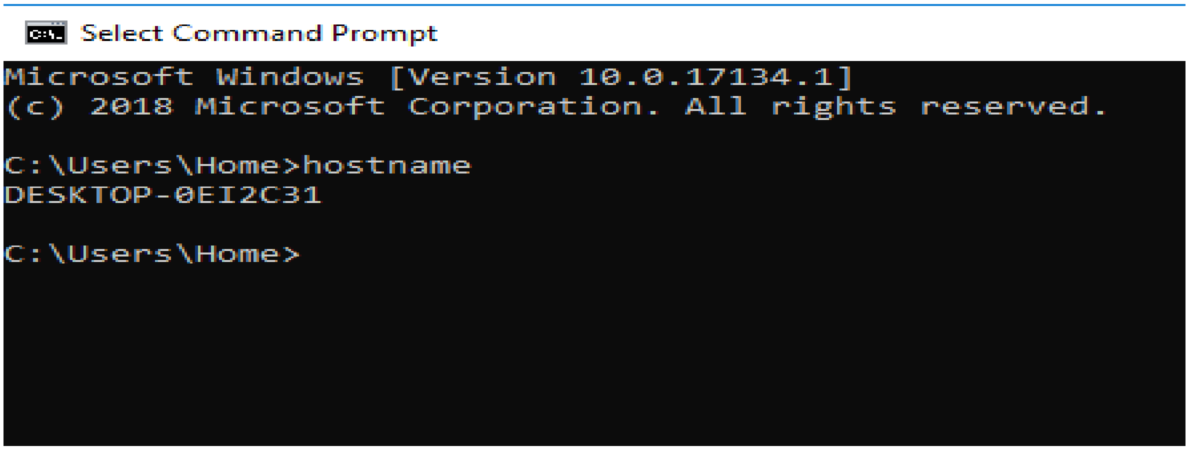


```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>PATHPING

Usage: pathping [-g host-list] [-h maximum_hops] [-i address] [-n]
                [-p period] [-q num_queries] [-w timeout]
                [-4] [-6] target_name

Options:
    -g host-list     Loose source route along host-list.
    -h maximum_hops  Maximum number of hops to search for target.
    -i address       Use the specified source address.
    -n               Do not resolve addresses to hostnames.
    -p period        Wait period milliseconds between pings.
    -q num_queries   Number of queries per hop.
    -w timeout       Wait timeout milliseconds for each reply.
    -4               Force using IPv4.
    -6               Force using IPv6.

C:\Users\Home>pathping -i
A value must be supplied for option -i.

C:\Users\Home>pathping 2 -g

Tracing route to 0.0.0.2 over a maximum of 30 hops

  0  No resources.
```

4. HOSTNAME

HOSTNAME: Displays the name of the machine

1) Screenshot of hostname command output



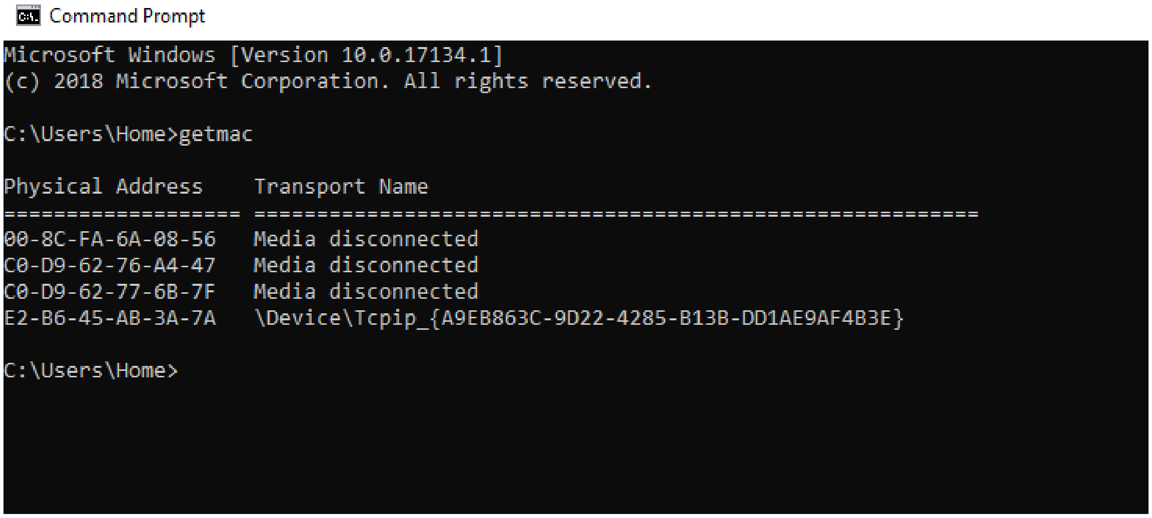
```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>hostname
DESKTOP-0EI2C31

C:\Users\Home>
```

5. GETMAC

This command enables an administrator to display the MAC address for one mor network adapters on a system.

Example: getmac

1) Screenshot of getmac command output



```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>getmac

Physical Address    Transport Name
=================== ==========================================================
00-8C-FA-6A-08-56   Media disconnected
C0-D9-62-76-A4-47   Media disconnected
C0-D9-62-77-6B-7F   Media disconnected
E2-B6-45-AB-3A-7A   \Device\Tcpip_{A9EB863C-9D22-4285-B13B-DD1AE9AF4B3E}

C:\Users\Home>
```

6. ARP

ARP: Resolving IP addresses to MAC addresses. Displays and modifies the translation tables of

IP addresses to physical addresses used by the ARP address resolution protocol.

ARP -s adr_inet adr_eth [adr_if]

ARP -d adr_inet [adr_if]

ARP -a [adr_inet] [-N adr_if]

- -a Displays active ARP entries by interrogating the current data protocol. If adr_inet is specified, only the physical and IP addresses of the specified computer are displayed. If more than one network interface uses ARP, entries for each ARP table are displayed.

- -g is the same as -a

- adr_inet Specifies an internet address.

- -N adr_if Displays ARP entries for the network interface specified by adr_if.

- -d Deletes the host specified by adr_inet.

- -s Adds the host and associates the adr_inet internet address with the adr_eth physical address. The physical address is given as 6 hexadecimal bytes separated by hyphens. The entry is permanent.

- adr_eth Specifies a physical address.

- adr_if Specifies the internet interface whose address translation table should be modified. When not specified, the first applicable interface will be used.

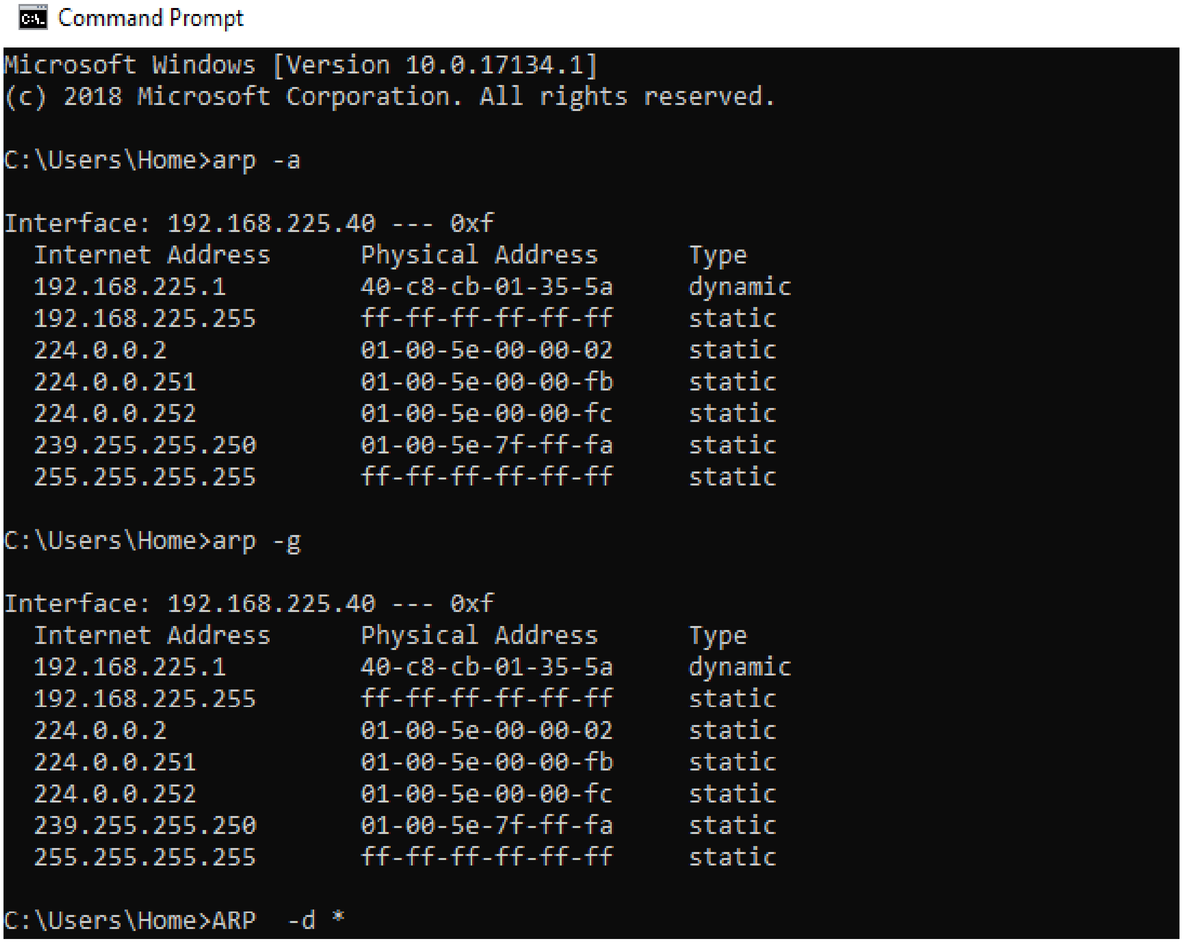
Examples

ARP –a

ARP -d *

ARP –g

1) Screenshot of arp command output



```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>arp -a

Interface: 192.168.225.40 --- 0xf
  Internet Address      Physical Address      Type
  192.168.225.1         40-c8-cb-01-35-5a     dynamic
  192.168.225.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\Home>arp -g

Interface: 192.168.225.40 --- 0xf
  Internet Address      Physical Address      Type
  192.168.225.1         40-c8-cb-01-35-5a     dynamic
  192.168.225.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\Home>ARP  -d *
```

7. TRACERT

TRACERT: Displays all intermediate IP addresses through which a packet passes through, between the local machine and the specified IP address.

tracert [IP address or domain name]

This command is useful if the ping command does return any data, to determine at what level the connection failed.

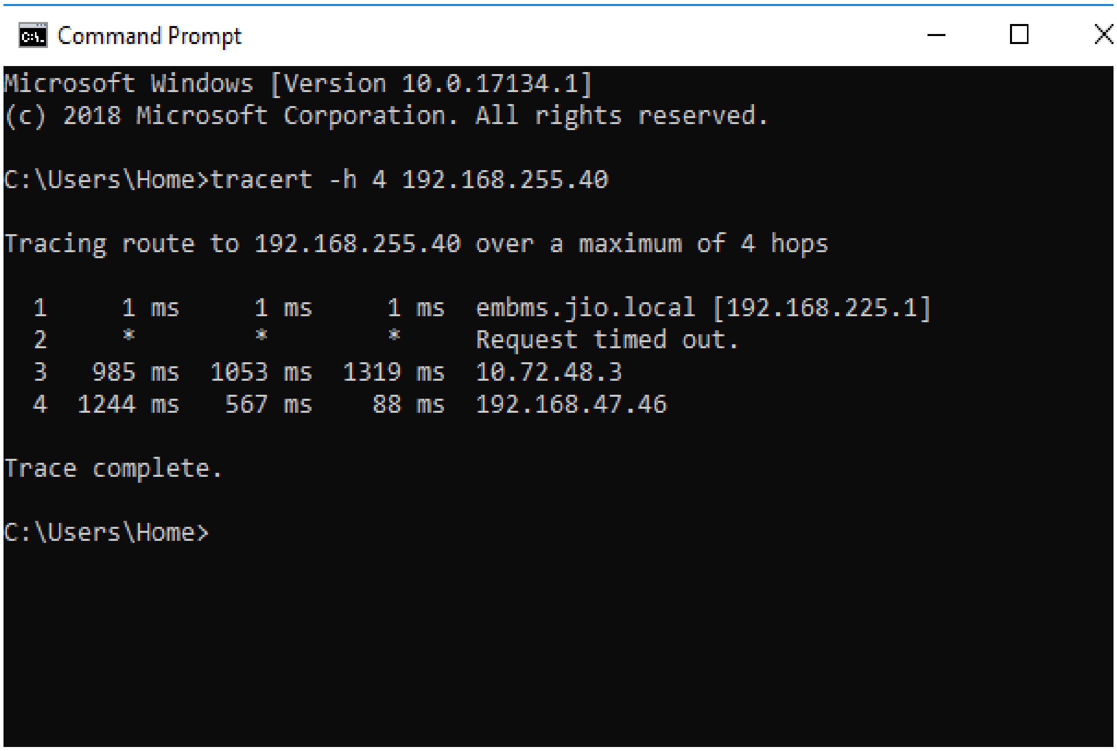
Examples:

TRACERT www.doubleclick.net

TRACERT 123.45.67.89

TRACERT local_server

1) Screenshot of tracert command output



```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>tracert -h 4 192.168.255.40

Tracing route to 192.168.255.40 over a maximum of 4 hops

  1     1 ms     1 ms     1 ms  embms.jio.local [192.168.225.1]
  2     *        *        *     Request timed out.
  3   985 ms  1053 ms  1319 ms  10.72.48.3
  4  1244 ms   567 ms    88 ms  192.168.47.46

Trace complete.

C:\Users\Home>
```

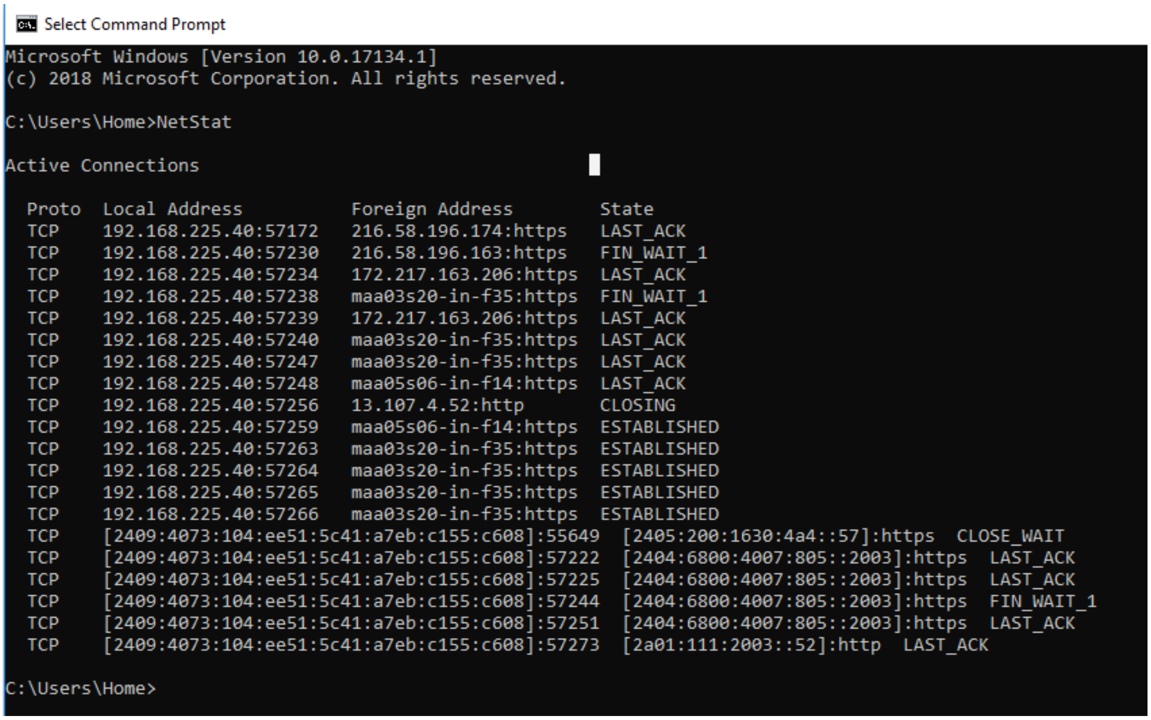
8. NETSTAT

NETSTAT: Displays the status of the TCP/IP stack on the local machine

NETSTAT [-a] [-e] [-n] [-s] [-p proto] [-r] [interval]

- -a Displays all connections and listening ports (server-side connections are normally inhibited).
- -e Displays Ethernet statistics. Can be combined with the -s option.
- -n Displays addresses and port numbers in numerical form.
- -p proto Shows connections for the protocol specified by proto, proto may be TCP or UDP. Used with the -s option to display per-protocol statistics, proto may be TCP, UDP or IP.
- -r Displays the contents of the routing table.
- -s Displays statistics by protocol. By default, statistics on TCP, UDP and IP are displayed, the -p option can be used to specify a subset.
- interval: Re-display the selected statistics, pausing after a specific "interval"(in seconds) between each display. Press Ctrl + C to stop displaying statistics.

1) Screenshot of netstat command output



```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>NetStat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    192.168.225.40:57172   216.58.196.174:https   LAST_ACK
  TCP    192.168.225.40:57230   216.58.196.163:https   FIN_WAIT_1
  TCP    192.168.225.40:57234   172.217.163.206:https  LAST_ACK
  TCP    192.168.225.40:57238   maa03s20-in-f35:https  FIN_WAIT_1
  TCP    192.168.225.40:57239   172.217.163.206:https  LAST_ACK
  TCP    192.168.225.40:57240   maa03s20-in-f35:https  LAST_ACK
  TCP    192.168.225.40:57247   maa03s20-in-f35:https  LAST_ACK
  TCP    192.168.225.40:57248   maa05s06-in-f14:https  LAST_ACK
  TCP    192.168.225.40:57256   13.107.4.52:http       CLOSING
  TCP    192.168.225.40:57259   maa05s06-in-f14:https  ESTABLISHED
  TCP    192.168.225.40:57263   maa03s20-in-f35:https  ESTABLISHED
  TCP    192.168.225.40:57264   maa03s20-in-f35:https  ESTABLISHED
  TCP    192.168.225.40:57265   maa03s20-in-f35:https  ESTABLISHED
  TCP    192.168.225.40:57266   maa03s20-in-f35:https  ESTABLISHED
  TCP    [2409:4073:104:ee51:5c41:a7eb:c155:c608]:55649  [2405:200:1630:4a4::57]:https  CLOSE_WAIT
  TCP    [2409:4073:104:ee51:5c41:a7eb:c155:c608]:57222  [2404:6800:4007:805::2003]:https  LAST_ACK
  TCP    [2409:4073:104:ee51:5c41:a7eb:c155:c608]:57225  [2404:6800:4007:805::2003]:https  LAST_ACK
  TCP    [2409:4073:104:ee51:5c41:a7eb:c155:c608]:57244  [2404:6800:4007:805::2003]:https  FIN_WAIT_1
  TCP    [2409:4073:104:ee51:5c41:a7eb:c155:c608]:57251  [2404:6800:4007:805::2003]:https  LAST_ACK
  TCP    [2409:4073:104:ee51:5c41:a7eb:c155:c608]:57273  [2a01:111:2003::52]:http  LAST_ACK

C:\Users\Home>
```

9. ROUTE

ROUTE: Displays or modifies the routing table

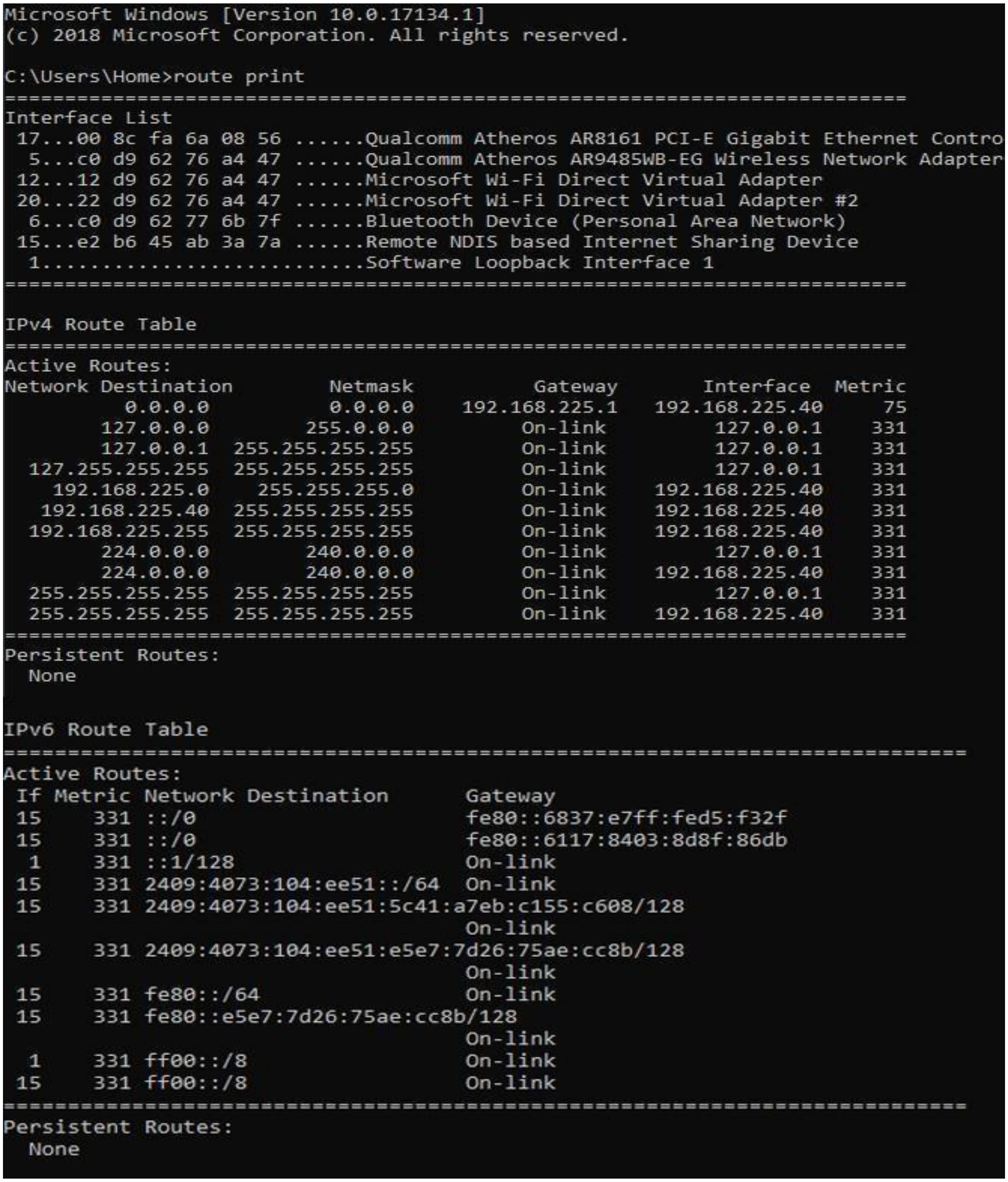
ROUTE [-f] [command [destination] [MASK network mask] [gateway]

- -f Clears the routing tables of all gateway entries. Used in conjunction with one of the below "commands", the tables are cleared before executing the command.
- -p Makes the entry into the table, residual (after reboot).

Specify one of four commands:

- DELETE: Deletes a route.
- PRINT: Displays a route.
- ADD: Adds a route.
- CHANGE: Modifies an existing route.
- destination: Specifies the host.
- MASK: If the MASK keyword is present, the next parameter is interpreted as the network mask parameter.
- netmask: Provided, it specifies the value of the subnet mask to be associated with this route entry. Unspecified, it takes the default value of 255.255.255.255.
- Gateway: Specifies the gateway.
- METRIC: Specifies the cost metric for the destination route print

1) Screenshot of route command output



```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>route print
===========================================================================
Interface List
 17...00 8c fa 6a 08 56 ......Qualcomm Atheros AR8161 PCI-E Gigabit Ethernet Contro
  5...c0 d9 62 76 a4 47 ......Qualcomm Atheros AR9485WB-EG Wireless Network Adapter
 12...12 d9 62 76 a4 47 ......Microsoft Wi-Fi Direct Virtual Adapter
 20...22 d9 62 76 a4 47 ......Microsoft Wi-Fi Direct Virtual Adapter #2
  6...c0 d9 62 77 6b 7f ......Bluetooth Device (Personal Area Network)
 15...e2 b6 45 ab 3a 7a ......Remote NDIS based Internet Sharing Device
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0    192.168.225.1   192.168.225.40     75
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
    192.168.225.0    255.255.255.0         On-link    192.168.225.40    331
   192.168.225.40  255.255.255.255         On-link    192.168.225.40    331
  192.168.225.255  255.255.255.255         On-link    192.168.225.40    331
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link    192.168.225.40    331
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link    192.168.225.40    331
===========================================================================
Persistent Routes:
  None

IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
 15    331 ::/0                     fe80::6837:e7ff:fed5:f32f
 15    331 ::/0                     fe80::6117:8403:8d8f:86db
  1    331 ::1/128                  On-link
 15    331 2409:4073:104:ee51::/64  On-link
 15    331 2409:4073:104:ee51:5c41:a7eb:c155:c608/128
                                    On-link
 15    331 2409:4073:104:ee51:e5e7:7d26:75ae:cc8b/128
                                    On-link
 15    331 fe80::/64                On-link
 15    331 fe80::e5e7:7d26:75ae:cc8b/128
                                    On-link
  1    331 ff00::/8                 On-link
 15    331 ff00::/8                 On-link
===========================================================================
Persistent Routes:
  None
```

10. NSLOOKUP (TCP/IP)

Lookup IP addresses, nameserver, mailserver, sartof authority, and inverse lookup on a NameServer.

Syntax

NSLOOKUP [-option] MyHost

NSLOOKUP [-option] MyHost MyNameServer (Lookup ip address of MyHost on MyNameServer)

Example:

nslookup google.com

nslookup –type=A google.com (Lookup IP addresses on a NameServer)

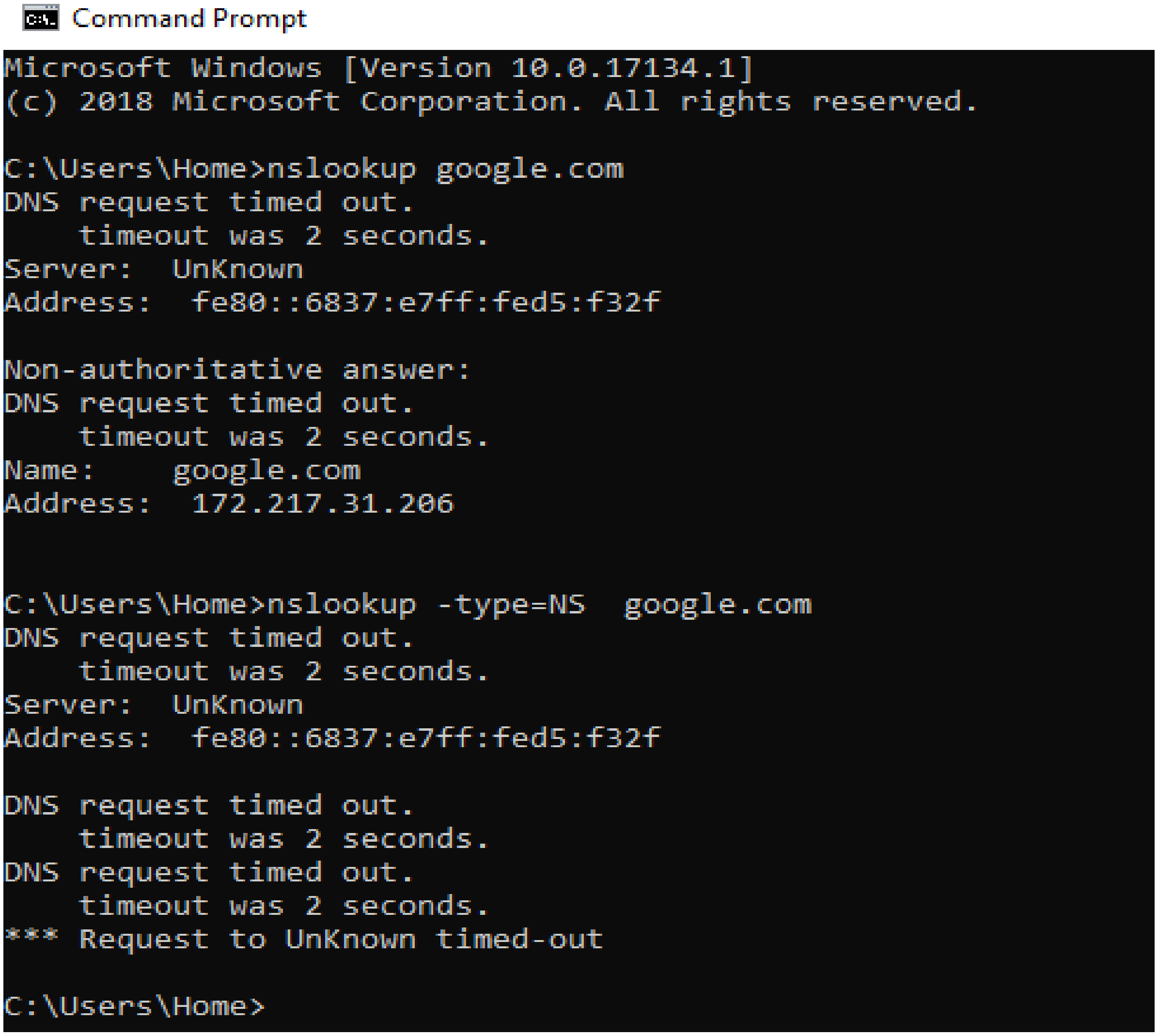
nslookup –type=MX google.com (Lookup Mail Server addresses on a NameServer

nslookup –type=SOA google.com (Lookup Start Of Authority details)

nslookup –type=NS google.com (Lookup Nameserver addresses of the domain name)

nslookup –type=ptr 1.1.168.192.in-addr.arpa (Inverse lookup)

1) Screenshot of nslookup command output

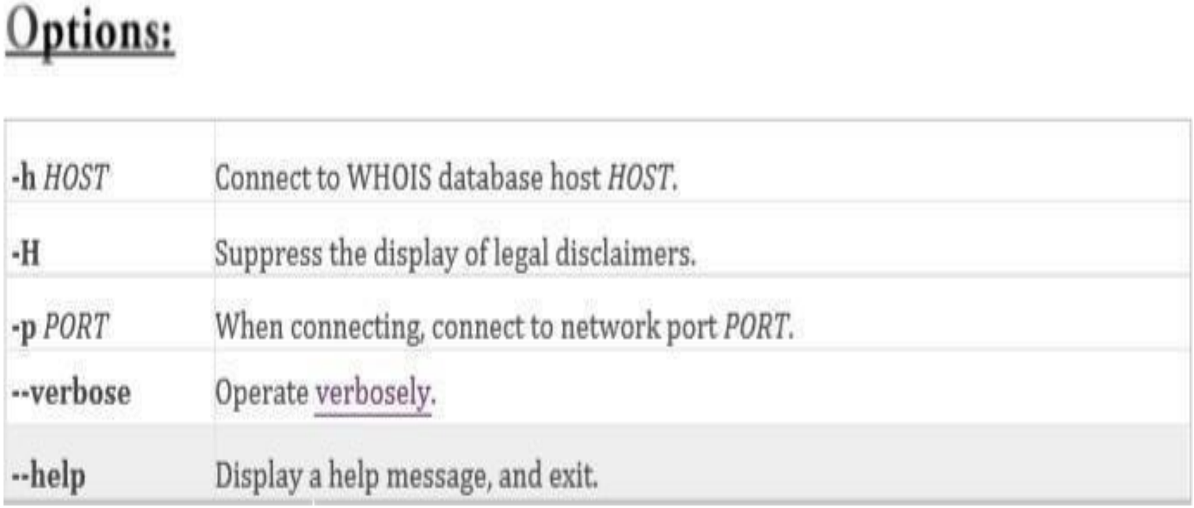


```
Microsoft Windows [Version 10.0.17134.1]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Home>nslookup google.com
DNS request timed out.
    timeout was 2 seconds.
Server:  UnKnown
Address:  fe80::6837:e7ff:fed5:f32f

Non-authoritative answer:
DNS request timed out.
    timeout was 2 seconds.
Name:    google.com
Address:  172.217.31.206


C:\Users\Home>nslookup -type=NS  google.com
DNS request timed out.
    timeout was 2 seconds.
Server:  UnKnown
Address:  fe80::6837:e7ff:fed5:f32f

DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
*** Request to UnKnown timed-out

C:\Users\Home>
```

11. WHOIS

whois searches for an object in a WHOIS database. WHOIS is a query and response protocol that is widely used for querying databases that store the registered users of an Internet resource, such as a domain name or an IP address block, but is also used for a wider range of other information.

Usage:whois [-h HOST] [-p PORT] [-aCFHlLMmrRSVx] [-g SOURCE:FIRST-LAST] [-i ATTR] [-S SOURCE] [-T TYPE] object



Options:

-h *HOST*	Connect to WHOIS database host *HOST*.
-H	Suppress the display of legal disclaimers.
-p *PORT*	When connecting, connect to network port *PORT*.
--verbose	Operate verbosely.
--help	Display a help message, and exit.

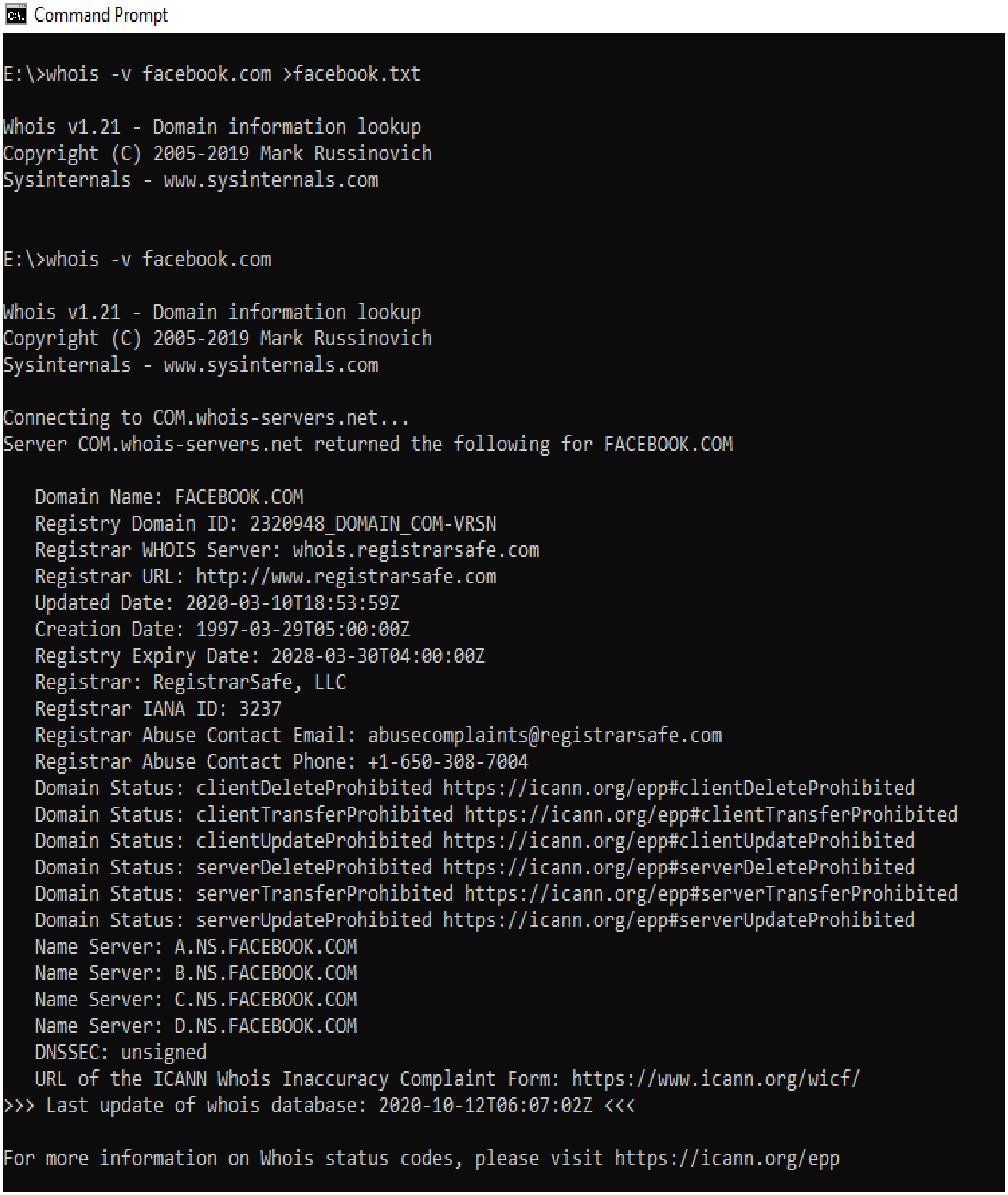
Command example

```
C:\ whois -v facebook.com

C:\ whois -v STAS.ORG
```

Copy the information as a text/xml file

```
C:\ whois -v facebook.com >facebook.txt

C:\ whois -v facebook.com >facebook.xml
```

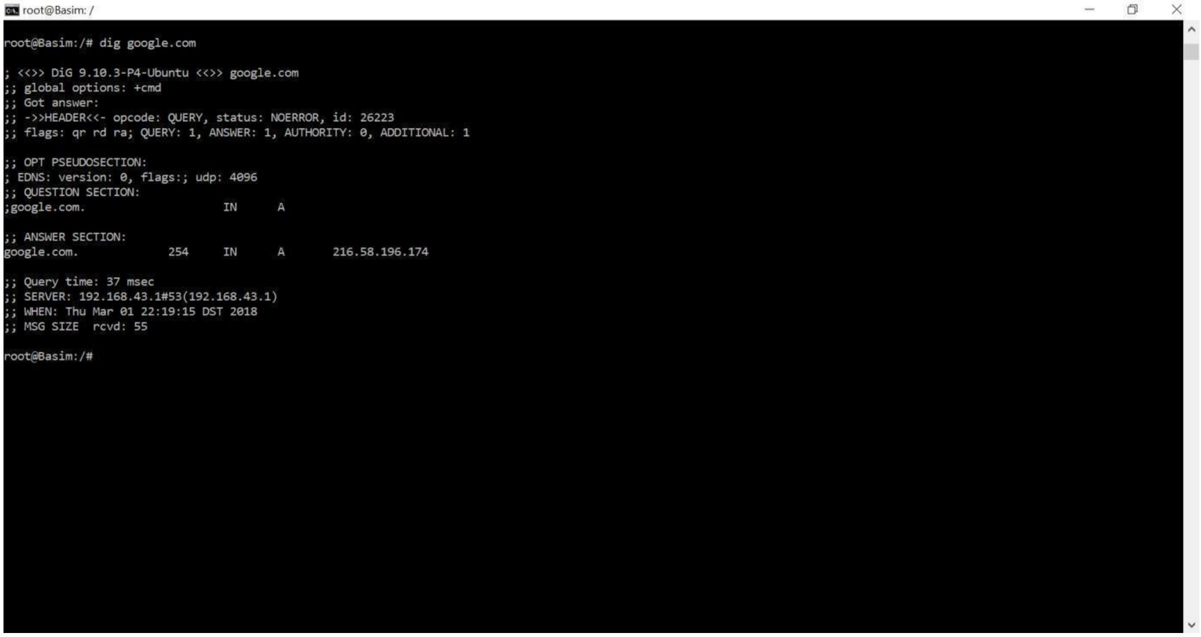
1) Screenshot of whois command output



```
E:\>whois -v facebook.com >facebook.txt

Whois v1.21 - Domain information lookup
Copyright (C) 2005-2019 Mark Russinovich
Sysinternals - www.sysinternals.com


E:\>whois -v facebook.com

Whois v1.21 - Domain information lookup
Copyright (C) 2005-2019 Mark Russinovich
Sysinternals - www.sysinternals.com

Connecting to COM.whois-servers.net...
Server COM.whois-servers.net returned the following for FACEBOOK.COM

   Domain Name: FACEBOOK.COM
   Registry Domain ID: 2320948_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.registrarsafe.com
   Registrar URL: http://www.registrarsafe.com
   Updated Date: 2020-03-10T18:53:59Z
   Creation Date: 1997-03-29T05:00:00Z
   Registry Expiry Date: 2028-03-30T04:00:00Z
   Registrar: RegistrarSafe, LLC
   Registrar IANA ID: 3237
   Registrar Abuse Contact Email: abusecomplaints@registrarsafe.com
   Registrar Abuse Contact Phone: +1-650-308-7004
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Name Server: A.NS.FACEBOOK.COM
   Name Server: B.NS.FACEBOOK.COM
   Name Server: C.NS.FACEBOOK.COM
   Name Server: D.NS.FACEBOOK.COM
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2020-10-12T06:07:02Z <<<

For more information on Whois status codes, please visit https://icann.org/epp
```

12. DIG

dig (domain information groper) is a network administration command-line tool for querying Domain Name System (DNS) servers.



```
root@Basim:/# dig google.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26223
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             254     IN      A       216.58.196.174

;; Query time: 37 msec
;; SERVER: 192.168.43.1#53(192.168.43.1)
;; WHEN: Thu Mar 01 22:19:15 DST 2018
;; MSG SIZE  rcvd: 55

root@Basim:/#
```

Usage: dig example.com

Result:

The various useful Windows- Network commands was executed and the output was verified successfully.

Experiment No:-02 Date: …/... /2024

Linux- Network Commands

1. HOSTNAME

Linux hostname command allows us to set and view the hostname of the system. A hostname is the name of any computer that is connected to a network that is uniquely identified over a network. It can be accessed without using a particular IP address. By default, the hostname of a system is set during the installation of OS.

Syntax: Hostname

The hostname command supports the following command-line options:

-a, --alias: It is used to display the alias name of the host (If it is defined). However, the '-a' option is deprecated.

-A, --all-fqdns: It is used to display the FQDNs of the system. The '-A' option lists all the configured network addresses on all interfaces and converts them to DNS names. It skips the addresses that can not be converted because they don't have proper reverse ip entry.

-b, --boot: It allows to set a hostname for always. If any hostname is not specified, it will use the default hostname (i.e., localhost).

-d, --domain: It is used to print the DNS domain name. Don't confuse between domainname and hostname command. The domainname command displays the NIS domain name, and it shows the DNS domain name.

-f, --fqdn, --long: It is used to print the FQDN (Fully Qualified Domain Name). An FQDN contains the short hostname and the DNS domain name. The FQDN and DNS domain name can be changed in the "/etc/hosts" file, except for the BIND or NIS host lookups.

-F, --file filename: It is used to read the hostname from a file.

-i, --ip-address: It is used to print the network addresses of the hostname. This option will work only if the hostname is in resolving circumstances.

-I, --all-ip-addresses: It is used to print the network addresses of the host. It will list all the configured addresses on all network interfaces. In this option, the loopback interface and Ipv6 local addresses are skipped. This option does depend on the name resolution like the '-' option.

2. PING

Linux ping command stands for (Packet Internet Groper). It checks connectivity between two nodes to see if a server is available. It sends ICMP ECHO_REQUEST packets to network hosts and displays the data on the remote server's response. It checks if a remote host is up, or that network interfaces can be reached. Further, it is used to check if a network connection is available between two devices. It is also handy tool for checking your network connection and verifying network issues.Ping command keeps executing and sends the packet until you interrupt.To stop the execution, press "**CTRL**+C" keys

Syntax: ping <option> <destination>

Options:

The ping command supports the following command-line options:

-4: It used to use IPv4 only.

-6: It is used to use IPv6 only.

-a: It is used for the audible ping.

-A: It is used for an adaptive ping.

-b: It is used to ping a broadcast address.

-B: It is used for not changing the source address of probes.

-c count: It is used to stop after sending count ECHO_REQUEST packets.

3. NETSTAT

Linux netstat command stands for **Network statistics**. It displays information about different interface statistics, including open sockets, routing tables, and connection information. Further, it can be used to displays all the socket connections (including TCP, UDP). Apart from connected sockets, it also displays the sockets that are pending for connections. It is a handy tool for network and system administrators.

Syntax: **netstat**

4. IFCONFIG

The command ifconfig stands for interface configurator. This command enables us to initialize an interface, assign IP address, enable or disable an interface. It display route and network interface.You can view IP address, MAC address and MTU (Maximum Transmission Unit) with ifconfig command.

Syntax: **ifconfig**

5. NSLOOKUP

This command is also used to find DNS related query.

Syntax: **nslookup <domainName>**

Experiment No:-03 Date: …/... /2024

Network analyzing tool &Packet capturing tool

Aim:

To familiarize working with monitor network communication using Wireshark.

Theoretical Background:

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, in May 2006 the project was renamed Wireshark due to trademark issues. Wireshark allows the user to put network interface controllers that support promiscuous mode into that mode, in order to see all traffic visible on that interface, not just traffic addressed to one of the interface's configured addresses and broadcast/multicast traffic. However, when capturing with a packet analyzer in promiscuous mode on a port on a network switch, not all of the traffic travelling through the switch will necessarily be sent to the port on which the capture is being done, so capturing in promiscuous mode will not necessarily be sufficient to see all traffic on the network. Port mirroring or various network taps extend capture to any point on the network. Simple passive taps are extremely resistant to tampering.

Purpose of using Wireshark

Here are some examples people use Wireshark for:

- Network administrators use it to troubleshoot network problems
- Network security engineers use it to examine security problems
- Developers use it to debug protocol implementations
- People use it to learn network protocol internals.

Features

The following are some of the many features Wireshark provides:

- Available for UNIX and Windows.
- Capture live packet data from a network interface.
- Open files containing packet data captured withtcpdump/WinDump, Wireshark, and a number of other packet capture programs.
- Import packets from text files containing hex dumps of packet data.
- Display packets with very detailed protocol information.
- Save packet data captured.
- Export some or all packets in a number of capture file formats.
- Filter packets on many criteria.
- Search for packets on many criteria.
- Colorize packet display based on filters.

Packet Capture (Packet Sniffing)

A packet sniffer is an application which can capture and analyse network traffic which is passing through a system's Network Interface Card (NIC). The sniffer sets the card to promiscuous mode which means all traffic is read, whether it is addressed to that machine or not. The figure below shows an attacker sniffing packets from the network, and the Wireshark packet sniffer/analyser (formerly known as ethereal).

Packet Analyzer

A packet analyzer is a computer program or a piece of computer hardware that can intercept and log traffic passing over a digital network or part of a network. As data streams flow across the network, the sniffer captures each packet and if needed decodes the packet's raw data, showing the values of various fields in the packet, and analyzes its content according to the appropriate RFC or other specifications.

Color Coding

You'll probably see packets highlighted in green, blue and black. Wireshark uses colors to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic and black identifies TCP packets with problems

Uses of packet analyzer

- Analyze network problems
- Detect network intrusion attempts
- Detect network misuse by internal and external users
- Documenting regulatory compliance through logging all perimeter and endpoint traffic
- Gain information for effecting a network intrusion
- Isolate exploited systems
- Gather and report network statistics
- Filter suspect content from network traffic
- Serve as primary data source for day-to-day network monitoring and management
- Debug client/server communications
- Debug network protocol implementations

Packet Analysis details:-

Wireshark is an open source cross-platform packet capture and analysis tool, with versions for Windows and Linux. The GUI window gives a detailed breakdown of the network protocol stack for each packet, colorising packet details based on protocol, as well as having functionality to filter and search the traffic, and pick out TCP streams. Wireshark can also save packet data to files for offline analysis and export/import packet captures to/from other tools. Statistics can also be generated for packet capture files. Wireshark can be used for **network troubleshooting**, to **investigate security issues**, and to **analyse and understand network protocols**.

The packet sniffer can exploit information passed in plaintext, i.e. not encrypted. Examples of **protocols** which pass information in plaintext are **Telnet, FTP, SNMP, POP, and HTTP**.

Wireshark is a GUI based network capture tool. There is a command line based version of the packet capture utility, called **TShark**. TShark provides many of the same features as it's big brother, but is console-based. It can be a good alternative if only command line access is available, and also uses less resources as it has no GUI to generate.

Using Wireshark to Capture Traffic

Select a Network Interface to Capture Packets through.

Start the Wireshark application. When Wireshark is first run, a default, or blank window is shown. To list the available network interfaces, select the **Capture->Interfaces** menu option.

To capture network traffic click the **Start** button for the network interface you want to capture traffic on. Windows can have a long list of virtual interfaces, before the Ethernet Network Interface Card (NIC).

Note: The total incoming packets, for each interface, are displayed in the column to the left of the **Start** buttons.

Generate some network traffic with a Web Browser, such as Internet Explorer or Chrome. Your Wireshark window should show the packets, and now look something like.

To stop the capture, select the **Capture->Stop** menu option, Ctrl+E, or the Stop toolbar button. What you have created is a Packet Capture or ***'pcap'***, which you can now view and analyse using the Wireshark interface, or save to disk to analyse later.

The capture is split into 3 parts:

1. **Packet List Panel –** this is a list of packets in the current capture. It colours the packets based on the protocol type. When a packet is selected, the details are shown in the two panels below.

2. **Packet Details Panel –** this shows the details of the selected packet. It shows the different protocols making up the layers of data for this packet. Layers include Frame, Ethernet, IP, TCP/UDP/ICMP, and application protocols such as HTTP.

3. **Packet Bytes Panel –** shows the packet bytes in Hex and ASCII encodings.

Search back through your capture, and find an **HTTP** packet containing a **GET** command. Click on the packet in the **Packet List Panel**. Then expand the HTTP layer in the **Packet Details Panel**, from the packet**.**

To select more detailed options when starting a capture, select the **Capture->Options** menu option, or **Ctrl+K**, or the Capture Options button on the toolbar (the wrench).

- ***Capture Options > Interface* -** Again the important thing is to select the correct Network Interface to capture traffic through.
- ***Capture Options > Capture File* –** useful to save a file of the packet capture in real time, in case of a system crash.
- ***Display Options > Update list of packets in real time* –** A display option, which should be checked if you want to view the capture as it happens (typically switched off to capture straight to a file, for later analysis).
- ***Name Resolution > MAC name resolution* –** resolves the first 3 bytes of the MAC Address, the Organisation Unique Identifier (OUI), which represents the Manufacturer of the Card.
- ***Name Resolution > Network name resolution* –** does a DNS lookup for the IP Addresses captured, to display the network name. Set to off by default, so covert scans do not generate this DNS traffic, and tip off who's packets you are sniffing.

Make sure the **MAC name resolution** is selected. Start the capture, and generate some Web traffic again, then stop the capture.

Install Wireshark in Ubuntu

First, we install Wireshark from the terminal

```
sudo apt-get install wireshark
```

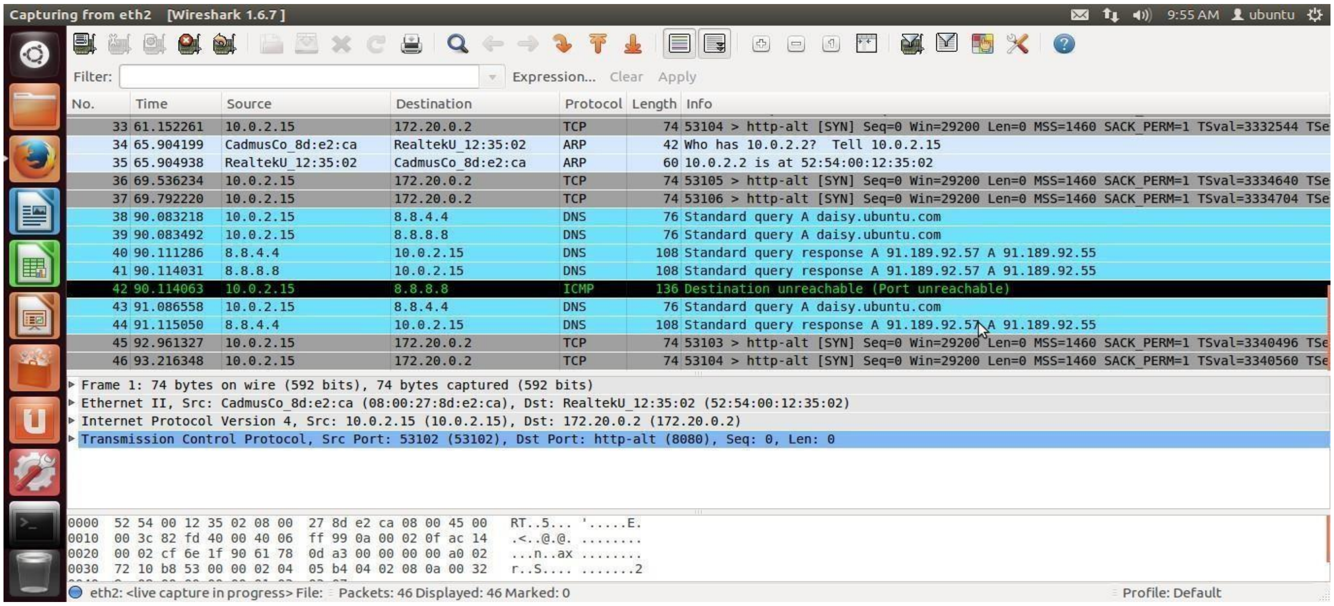
Install Wireshark on Windows 8/10 .

- Download the install file from http://www.wireshark.org/download.html
- Right click on the install file and click on properties

- In the properties window, click on the compatibility tab and change the option for “Run this program in compatibility mode for” to “Windows 8/10 ” and click on OK.
- Right click on the install and click on “Run as Administrator”
- Make sure you choose the option to install the NPF as a service during the install prompts. This will allow all users on the machine to use Wireshark without admin privileges.

Working with Wireshark

1) Screenshot of performing packet capturing



No.	Time	Source	Destination	Protocol	Length	Info
33	61.152261	10.0.2.15	172.20.0.2	TCP	74	53104 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3332544 TSe
34	65.904199	CadmusCo_8d:e2:ca	RealtekU_12:35:02	ARP	42	Who has 10.0.2.2? Tell 10.0.2.15
35	65.904938	RealtekU_12:35:02	CadmusCo_8d:e2:ca	ARP	60	10.0.2.2 is at 52:54:00:12:35:02
36	69.536234	10.0.2.15	172.20.0.2	TCP	74	53105 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3334640 TSe
37	69.792220	10.0.2.15	172.20.0.2	TCP	74	53106 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3334704 TSe
38	90.083218	10.0.2.15	8.8.4.4	DNS	76	Standard query A daisy.ubuntu.com
39	90.083492	10.0.2.15	8.8.8.8	DNS	76	Standard query A daisy.ubuntu.com
40	90.111286	8.8.4.4	10.0.2.15	DNS	108	Standard query response A 91.189.92.57 A 91.189.92.55
41	90.114031	8.8.8.8	10.0.2.15	DNS	108	Standard query response A 91.189.92.57 A 91.189.92.55
42	90.114063	10.0.2.15	8.8.8.8	ICMP	136	Destination unreachable (Port unreachable)
43	91.086558	10.0.2.15	8.8.4.4	DNS	76	Standard query A daisy.ubuntu.com
44	91.115050	8.8.4.4	10.0.2.15	DNS	108	Standard query response A 91.189.92.57 A 91.189.92.55
45	92.961327	10.0.2.15	172.20.0.2	TCP	74	53103 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3340496 TSe
46	93.216348	10.0.2.15	172.20.0.2	TCP	74	53104 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3340560 TSe

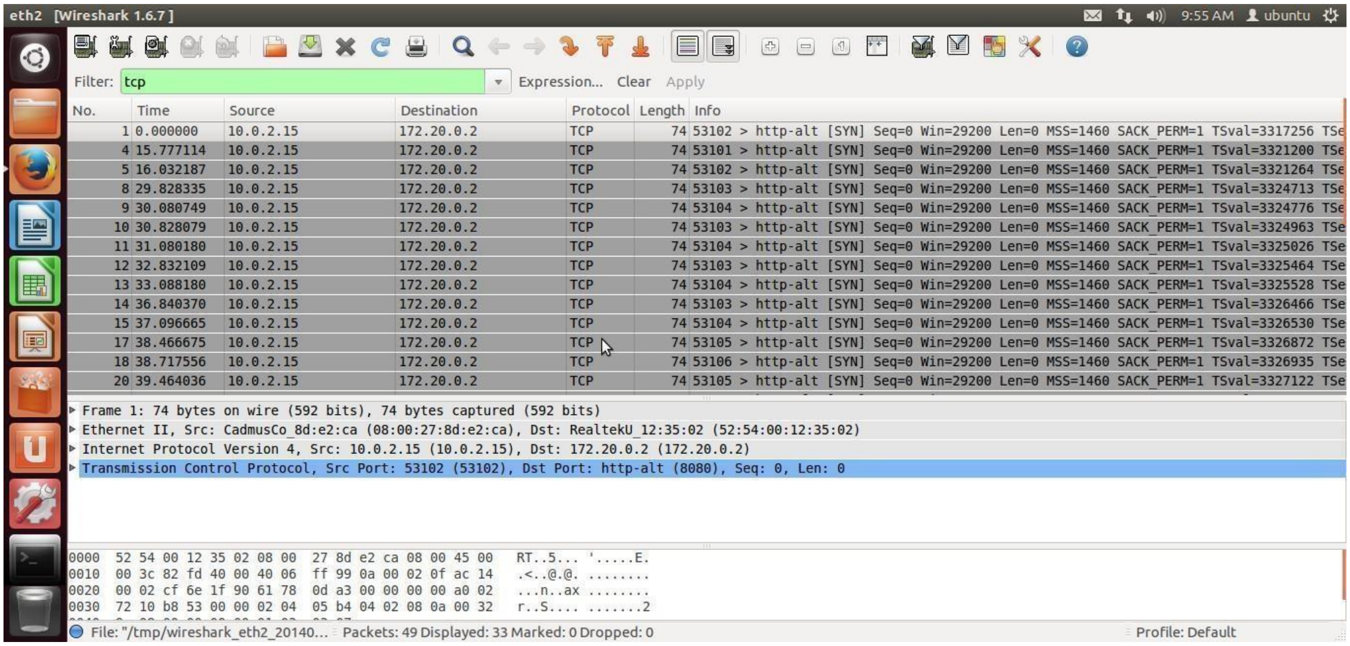
```
▸ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
▸ Ethernet II, Src: CadmusCo_8d:e2:ca (08:00:27:8d:e2:ca), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 172.20.0.2 (172.20.0.2)
▸ Transmission Control Protocol, Src Port: 53102 (53102), Dst Port: http-alt (8080), Seq: 0, Len: 0
```

```
0000  52 54 00 12 35 02 08 00  27 8d e2 ca 08 00 45 00   RT..5... '.....E.
0010  00 3c 82 fd 40 00 40 06  ff 99 0a 00 02 0f ac 14   .<..@.@. ........
0020  00 02 cf 6e 1f 90 61 78  0d a3 00 00 00 00 a0 02   ...n..ax ........
0030  72 10 b8 53 00 00 02 04  05 b4 04 02 08 0a 00 32   r..S.... .......2
```

eth2: <live capture in progress> File: Packets: 46 Displayed: 46 Marked: 0 Profile: Default

2) Screenshot of filtering TCP packets.

Perform live packet filtering and save the filtered packet. For that type 'tcp' on the filter tab and all the tcp packets will be filtered out.

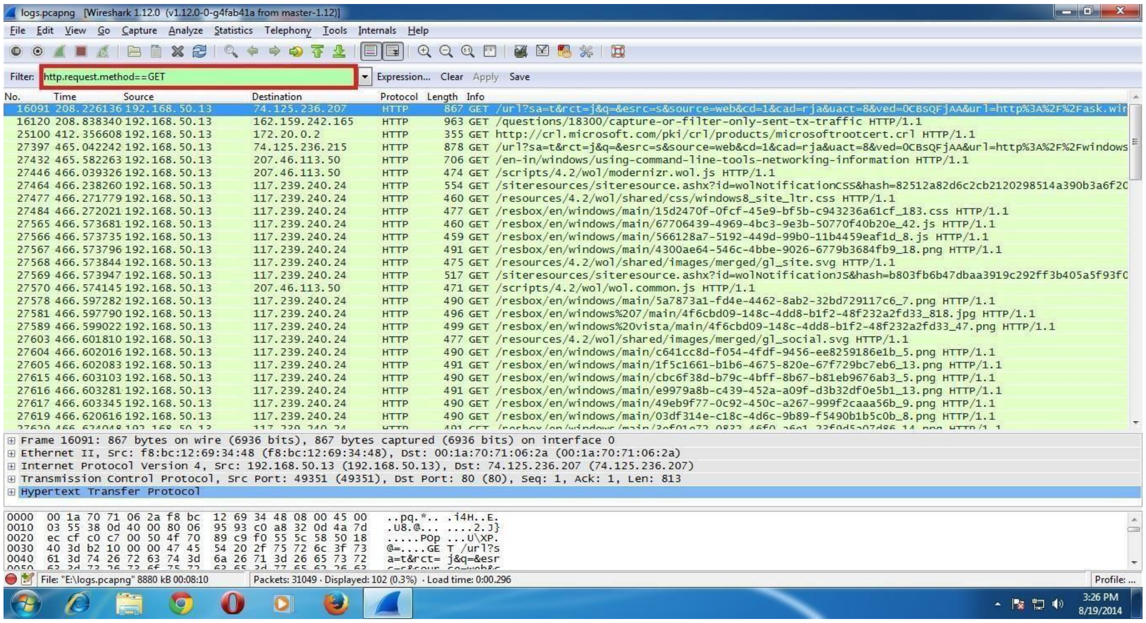


Filter: tcp

```
No.  Time       Source     Destination  Protocol Length Info
   1 0.000000   10.0.2.15  172.20.0.2   TCP          74 53102 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3317256 TSe
   4 15.777114  10.0.2.15  172.20.0.2   TCP          74 53101 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3321200 TSe
   5 16.032187  10.0.2.15  172.20.0.2   TCP          74 53102 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3321264 TSe
   8 29.828335  10.0.2.15  172.20.0.2   TCP          74 53103 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3324713 TSe
   9 30.080749  10.0.2.15  172.20.0.2   TCP          74 53104 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3324776 TSe
  10 30.828079  10.0.2.15  172.20.0.2   TCP          74 53103 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3324963 TSe
  11 31.080180  10.0.2.15  172.20.0.2   TCP          74 53104 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3325026 TSe
  12 32.832109  10.0.2.15  172.20.0.2   TCP          74 53103 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3325464 TSe
  13 33.088180  10.0.2.15  172.20.0.2   TCP          74 53104 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3325528 TSe
  14 36.840370  10.0.2.15  172.20.0.2   TCP          74 53103 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3326466 TSe
  15 37.096665  10.0.2.15  172.20.0.2   TCP          74 53104 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3326530 TSe
  17 38.466675  10.0.2.15  172.20.0.2   TCP          74 53105 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3326872 TSe
  18 38.717556  10.0.2.15  172.20.0.2   TCP          74 53106 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3326935 TSe
  20 39.464036  10.0.2.15  172.20.0.2   TCP          74 53105 > http-alt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3327122 TSe

▸ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
▸ Ethernet II, Src: CadmusCo_8d:e2:ca (08:00:27:8d:e2:ca), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 172.20.0.2 (172.20.0.2)
▸ Transmission Control Protocol, Src Port: 53102 (53102), Dst Port: http-alt (8080), Seq: 0, Len: 0

0000  52 54 00 12 35 02 08 00  27 8d e2 ca 08 00 45 00   RT..5... '.....E.
0010  00 3c 82 fd 40 00 40 06  ff 99 0a 00 02 0f ac 14   .<..@.@. ........
0020  00 02 cf 6e 1f 90 61 78  0d a3 00 00 00 00 a0 02   ...n..ax ........
0030  72 10 b8 53 00 00 02 04  05 b4 04 02 08 0a 00 32   r..S.... .......2

File: "/tmp/wireshark_eth2_20140... Packets: 49 Displayed: 33 Marked: 0 Dropped: 0     Profile: Default
```

3) Screenshot of filtering http packet request

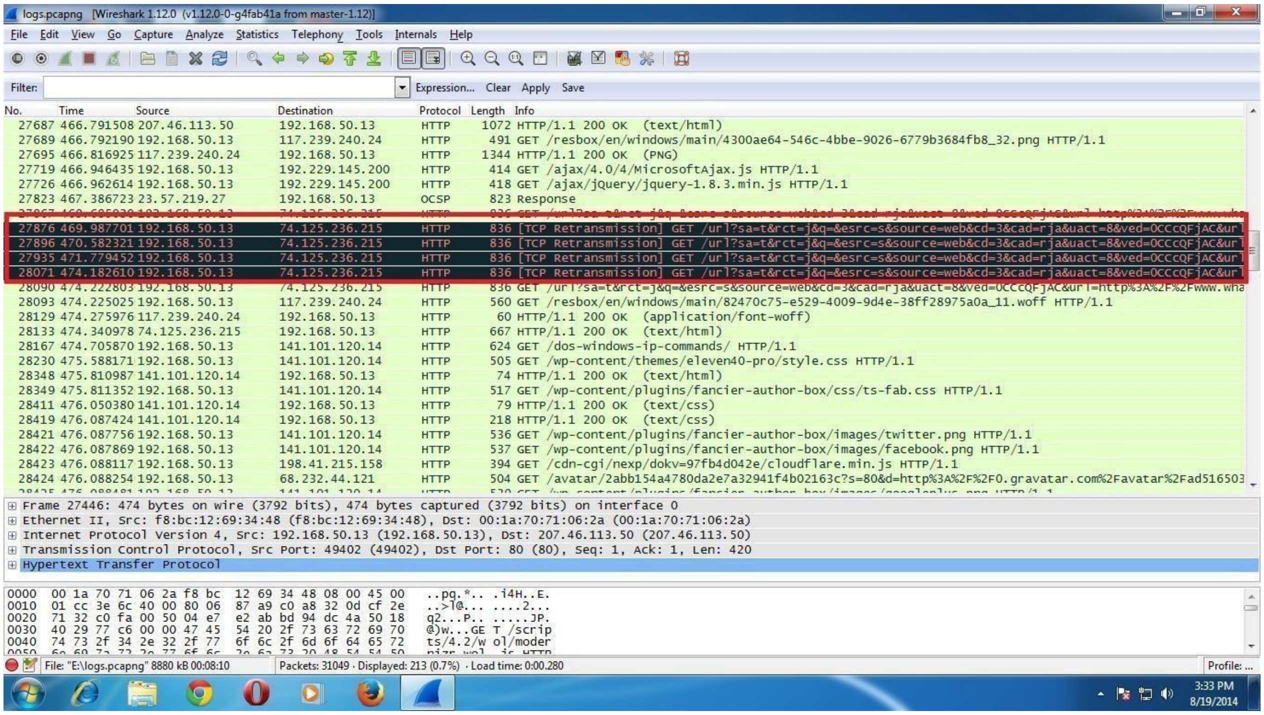
Type "http.request.method==GET" in filter bar.



Filter: http.request.method==GET

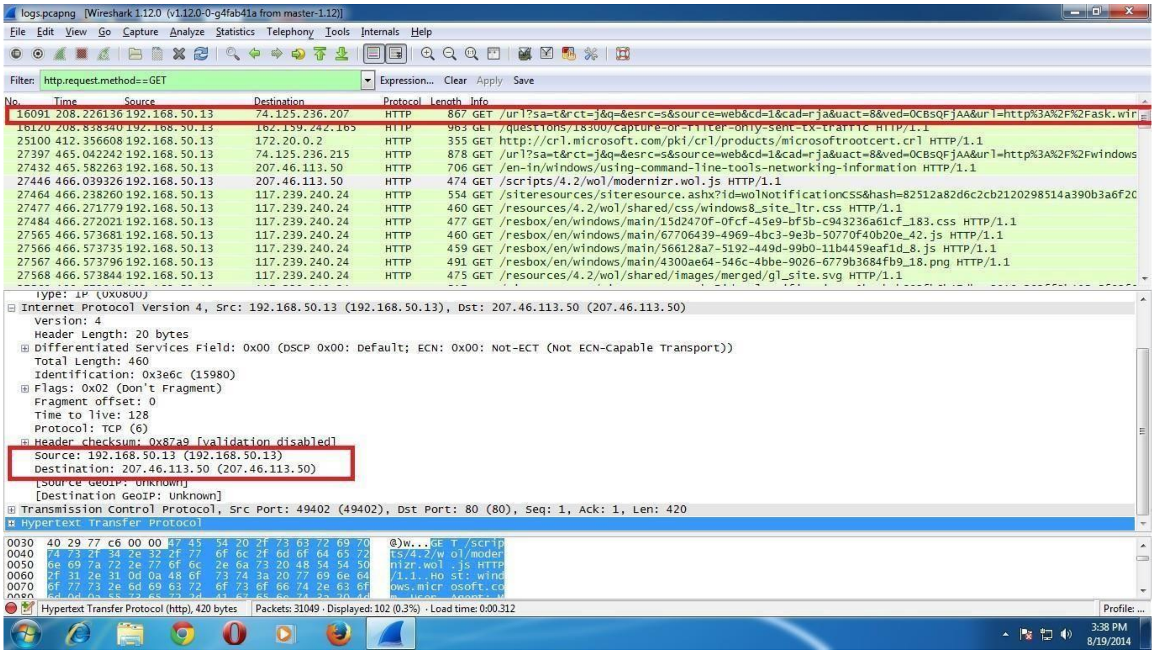
```
No.   Time       Source         Destination      Protocol Length Info
16091 208.226136 192.168.50.13  74.125.236.207   HTTP     867 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fask.wir
16120 208.838340 192.168.50.13  162.159.242.165  HTTP     963 GET /questions/18300/capture-or-filter-only-sent-tx-traffic HTTP/1.1
25100 412.356608 192.168.50.13  172.20.0.2       HTTP     355 GET http://crl.microsoft.com/pki/crl/products/microsoftrootcert.crl HTTP/1.1
27397 465.042242 192.168.50.13  74.125.236.215   HTTP     878 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fwindows
27432 465.582263 192.168.50.13  207.46.113.50    HTTP     706 GET /en-in/windows/using-command-line-tools-networking-information HTTP/1.1
27446 466.039326 192.168.50.13  207.46.113.50    HTTP     474 GET /scripts/4.2/wol/modernizr.wol.js HTTP/1.1
27464 466.238260 192.168.50.13  117.239.240.24   HTTP     554 GET /siteresources/siteresource.ashx?id=wolNotificationCSS&hash=82512a82d6c2cb2120298514a390b3a6f2C
27477 466.271779 192.168.50.13  117.239.240.24   HTTP     460 GET /resources/4.2/wol/shared/css/windows8_site_ltr.css HTTP/1.1
27484 466.272021 192.168.50.13  117.239.240.24   HTTP     477 GET /resbox/en/windows/main/15d2470f-0fcf-45e9-bf5b-c943236a61cf_183.css HTTP/1.1
27565 466.573681 192.168.50.13  117.239.240.24   HTTP     460 GET /resbox/en/windows/main/67706439-4969-4bc3-9e3b-50770f40b20e_42.js HTTP/1.1
27566 466.573735 192.168.50.13  117.239.240.24   HTTP     459 GET /resbox/en/windows/main/566128a7-5192-449d-99b0-11b4459eaf1d_8.js HTTP/1.1
27567 466.573796 192.168.50.13  117.239.240.24   HTTP     491 GET /resbox/en/windows/main/4300ae64-546c-4bbe-9026-6779b3684fb9_18.png HTTP/1.1
27568 466.573844 192.168.50.13  117.239.240.24   HTTP     475 GET /resources/4.2/wol/shared/images/merged/gl_site.svg HTTP/1.1
27569 466.573947 192.168.50.13  117.239.240.24   HTTP     517 GET /siteresources/siteresource.ashx?id=wolNotificationJS&hash=b803fb6b47dbaa3919c292ff3b405a5f93fC
27570 466.574145 192.168.50.13  207.46.113.50    HTTP     471 GET /scripts/4.2/wol/wol.common.js HTTP/1.1
27578 466.597282 192.168.50.13  117.239.240.24   HTTP     490 GET /resbox/en/windows/main/5a7873a1-fd4e-4462-8ab2-32bd729117c6_7.png HTTP/1.1
27581 466.597790 192.168.50.13  117.239.240.24   HTTP     496 GET /resbox/en/windows%207/main/4f6cbd09-148c-4dd8-b1f2-48f232a2fd33_818.jpg HTTP/1.1
27589 466.599022 192.168.50.13  117.239.240.24   HTTP     499 GET /resbox/en/windows%20vista/main/4f6cbd09-148c-4dd8-b1f2-48f232a2fd33_47.png HTTP/1.1
27603 466.601810 192.168.50.13  117.239.240.24   HTTP     477 GET /resources/4.2/wol/shared/images/merged/gl_social.svg HTTP/1.1
27604 466.602016 192.168.50.13  117.239.240.24   HTTP     490 GET /resbox/en/windows/main/c641cc8d-f054-4fdf-9456-ee8259186e1b_5.png HTTP/1.1
27605 466.602083 192.168.50.13  117.239.240.24   HTTP     491 GET /resbox/en/windows/main/1f5c1661-b1b6-4675-820e-67f729bc7eb6_13.png HTTP/1.1
27615 466.603103 192.168.50.13  117.239.240.24   HTTP     490 GET /resbox/en/windows/main/cbc6f38d-b79c-4bff-8b67-b81eb9676ab3_5.png HTTP/1.1
27616 466.603281 192.168.50.13  117.239.240.24   HTTP     491 GET /resbox/en/windows/main/e9979a8b-c439-452a-a09f-d3b32df0e5b1_13.png HTTP/1.1
27617 466.603345 192.168.50.13  117.239.240.24   HTTP     490 GET /resbox/en/windows/main/49eb9f77-0c92-450c-a267-999f2caaa56b_9.png HTTP/1.1
27619 466.620616 192.168.50.13  117.239.240.24   HTTP     490 GET /resbox/en/windows/main/03df314e-c18c-4d6c-9b89-f5490b1b5c0b_8.png HTTP/1.1

⊞ Frame 16091: 867 bytes on wire (6936 bits), 867 bytes captured (6936 bits) on interface 0
⊞ Ethernet II, Src: f8:bc:12:69:34:48 (f8:bc:12:69:34:48), Dst: 00:1a:70:71:06:2a (00:1a:70:71:06:2a)
⊞ Internet Protocol Version 4, Src: 192.168.50.13 (192.168.50.13), Dst: 74.125.236.207 (74.125.236.207)
⊞ Transmission Control Protocol, Src Port: 49351 (49351), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 813
⊞ Hypertext Transfer Protocol

0000  00 1a 70 71 06 2a f8 bc  12 69 34 48 08 00 45 00   ..pq.*.. .i4H..E.
0010  03 55 38 0d 40 00 80 06  95 93 c0 a8 32 0d 4a 7d   .U8.@... ....2.J}
0020  ec cf c0 c7 00 50 4f 70  89 c9 f0 55 5c 58 50 18   .....POp ...U\XP.
0030  40 3d b2 10 00 00 47 45  54 20 2f 75 72 6c 3f 73   @=....GE T /url?s
0040  61 3d 74 26 72 63 74 3d  6a 26 71 3d 26 65 73 72   a=t&rct= j&q=&esr

File: "E:\logs.pcapng" 8880 kB 00:08:10   Packets: 31049 · Displayed: 102 (0.3%) · Load time: 0:00.296   Profile: ...
```

4) Screenshot showing retransmission of corrupted packet.


logs.pcapng [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.   Time       Source          Destination      Protocol Length Info
27687 466.791508 207.46.113.50   192.168.50.13    HTTP     1072 HTTP/1.1 200 OK  (text/html)
27689 466.792190 192.168.50.13   117.239.240.24   HTTP      491 GET /resbox/en/windows/main/4300ae64-546c-4bbe-9026-6779b3684fb8_32.png HTTP/1.1
27695 466.816925 117.239.240.24  192.168.50.13    HTTP     1344 HTTP/1.1 200 OK  (PNG)
27719 466.946435 192.168.50.13   192.229.145.200  HTTP      414 GET /ajax/4.0/4/MicrosoftAjax.js HTTP/1.1
27726 466.962614 192.168.50.13   192.229.145.200  HTTP      418 GET /ajax/jQuery/jquery-1.8.3.min.js HTTP/1.1
27823 467.386723 23.57.219.27    192.168.50.13    OCSP      823 Response
27876 469.987701 192.168.50.13   74.125.236.215   HTTP      836 [TCP Retransmission] GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CCcQFjAC&url
27896 470.582321 192.168.50.13   74.125.236.215   HTTP      836 [TCP Retransmission] GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CCcQFjAC&url
27935 471.779452 192.168.50.13   74.125.236.215   HTTP      836 [TCP Retransmission] GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CCcQFjAC&url
28071 474.182610 192.168.50.13   74.125.236.215   HTTP      836 [TCP Retransmission] GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CCcQFjAC&url
28090 474.222803 192.168.50.13   74.125.236.215   HTTP      836 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CCcQFjAC&url=http%3A%2F%2Fwww.wha
28093 474.225025 192.168.50.13   117.239.240.24   HTTP      560 GET /resbox/en/windows/main/82470c75-e529-4009-9d4e-38ff28975a0a_11.woff HTTP/1.1
28129 474.275976 117.239.240.24  192.168.50.13    HTTP       60 HTTP/1.1 200 OK  (application/font-woff)
28133 474.340978 74.125.236.215  192.168.50.13    HTTP      667 HTTP/1.1 200 OK  (text/html)
28167 474.705870 192.168.50.13   141.101.120.14   HTTP      624 GET /dos-windows-ip-commands/ HTTP/1.1
28230 475.588171 192.168.50.13   141.101.120.14   HTTP      505 GET /wp-content/themes/eleven40-pro/style.css HTTP/1.1
28348 475.810987 141.101.120.14  192.168.50.13    HTTP       74 HTTP/1.1 200 OK  (text/html)
28349 475.811352 192.168.50.13   141.101.120.14   HTTP      517 GET /wp-content/plugins/fancier-author-box/css/ts-fab.css HTTP/1.1
28411 476.050380 141.101.120.14  192.168.50.13    HTTP       79 HTTP/1.1 200 OK  (text/css)
28419 476.087424 141.101.120.14  192.168.50.13    HTTP      218 HTTP/1.1 200 OK  (text/css)
28421 476.087756 192.168.50.13   141.101.120.14   HTTP      536 GET /wp-content/plugins/fancier-author-box/images/twitter.png HTTP/1.1
28422 476.087869 192.168.50.13   141.101.120.14   HTTP      537 GET /wp-content/plugins/fancier-author-box/images/facebook.png HTTP/1.1
28423 476.088117 192.168.50.13   198.41.215.158   HTTP      394 GET /cdn-cgi/nexp/dokv=97fb4d042e/cloudflare.min.js HTTP/1.1
28424 476.088254 192.168.50.13   68.232.44.121    HTTP      504 GET /avatar/2abb154a4780da2e7a32941f4b02163c?s=80&d=http%3A%2F%2F0.gravatar.com%2Favatar%2Fad516503

⊞ Frame 27446: 474 bytes on wire (3792 bits), 474 bytes captured (3792 bits) on interface 0
⊞ Ethernet II, Src: f8:bc:12:69:34:48 (f8:bc:12:69:34:48), Dst: 00:1a:70:71:06:2a (00:1a:70:71:06:2a)
⊞ Internet Protocol Version 4, Src: 192.168.50.13 (192.168.50.13), Dst: 207.46.113.50 (207.46.113.50)
⊞ Transmission Control Protocol, Src Port: 49402 (49402), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 420
⊞ Hypertext Transfer Protocol

0000  00 1a 70 71 06 2a f8 bc  12 69 34 48 08 00 45 00   ..pq.*.. .i4H..E.
0010  01 cc 3e 6c 40 00 80 06  87 a9 c0 a8 32 0d cf 2e   ..>l@... ....2...
0020  71 32 c0 fa 00 50 04 e7  e2 ab bd 94 dc 4a 50 18   q2...P.. .....JP.
0030  40 29 77 c6 00 00 47 45  54 20 2f 73 63 72 69 70   @)w...GE T /scrip
0040  74 73 2f 34 2e 32 2f 77  6f 6c 2f 6d 6f 64 65 72   ts/4.2/w ol/moder

File: "E:\logs.pcapng" 8880 kB 00:08:10   Packets: 31049 · Displayed: 213 (0.7%) · Load time: 0:00.280   Profile: ...
```

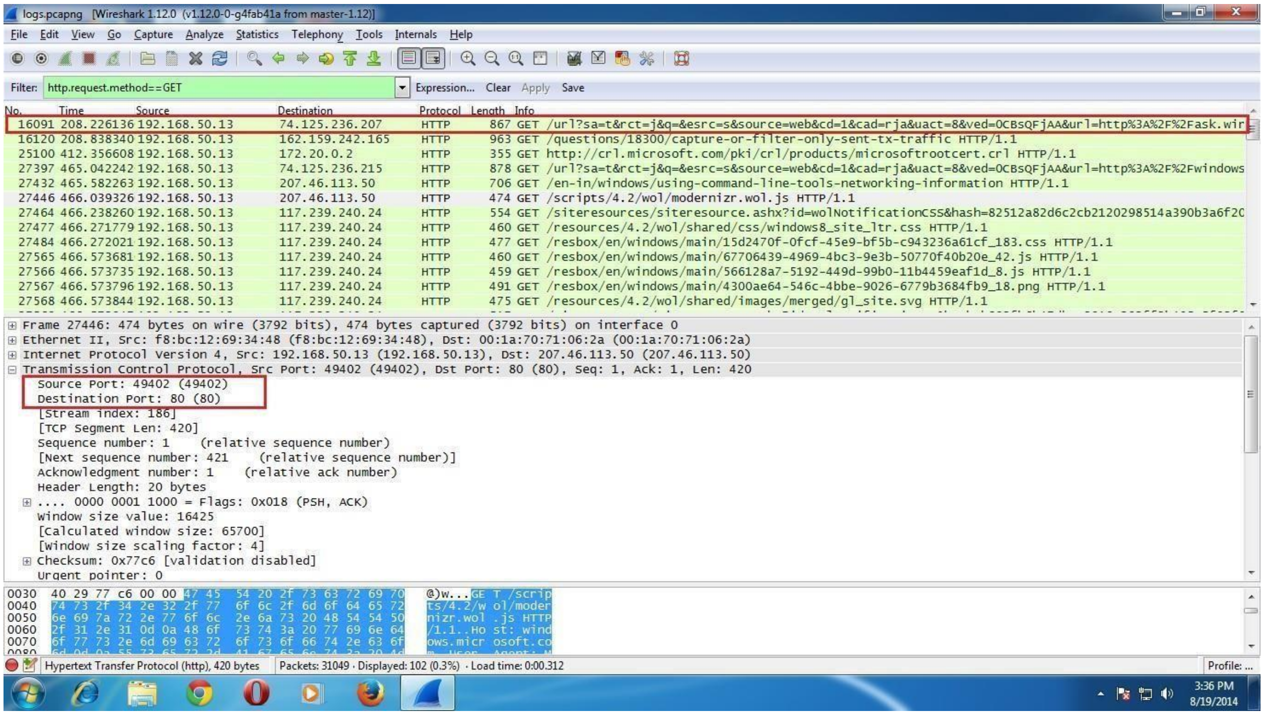
5) Screenshot showing source and destination IP addresses.


logs.pcapng [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

Filter: http.request.method==GET

```
No.   Time       Source          Destination      Protocol Length Info
16091 208.226136 192.168.50.13   74.125.236.207   HTTP      867 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fask.wir
16120 208.838340 192.168.50.13   162.159.242.165  HTTP      963 GET /questions/18300/capture-or-filter-only-sent-tx-traffic HTTP/1.1
25100 412.356608 192.168.50.13   172.20.0.2       HTTP      355 GET http://crl.microsoft.com/pki/crl/products/microsoftrootcert.crl HTTP/1.1
27397 465.042242 192.168.50.13   74.125.236.215   HTTP      878 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fwindows
27432 465.582263 192.168.50.13   207.46.113.50    HTTP      706 GET /en-in/windows/using-command-line-tools-networking-information HTTP/1.1
27446 466.039326 192.168.50.13   207.46.113.50    HTTP      474 GET /scripts/4.2/wol/modernizr.wol.js HTTP/1.1
27464 466.238260 192.168.50.13   117.239.240.24   HTTP      554 GET /siteresources/siteresource.ashx?id=wolNotificationCSS&hash=82512a82d6c2cb2120298514a390b3a6f20
27477 466.271779 192.168.50.13   117.239.240.24   HTTP      460 GET /resources/4.2/wol/shared/css/windows8_site_ltr.css HTTP/1.1
27484 466.272021 192.168.50.13   117.239.240.24   HTTP      477 GET /resbox/en/windows/main/15d2470f-0fcf-45e9-bf5b-c943236a61cf_183.css HTTP/1.1
27565 466.573681 192.168.50.13   117.239.240.24   HTTP      460 GET /resbox/en/windows/main/67706439-4969-4bc3-9e3b-50770f40b20e_42.js HTTP/1.1
27566 466.573735 192.168.50.13   117.239.240.24   HTTP      459 GET /resbox/en/windows/main/566128a7-5192-449d-99b0-11b4459eaf1d_8.js HTTP/1.1
27567 466.573796 192.168.50.13   117.239.240.24   HTTP      491 GET /resbox/en/windows/main/4300ae64-546c-4bbe-9026-6779b3684fb9_18.png HTTP/1.1
27568 466.573844 192.168.50.13   117.239.240.24   HTTP      475 GET /resources/4.2/wol/shared/images/merged/gl_site.svg HTTP/1.1

    Type: IP (0x0800)
⊟ Internet Protocol Version 4, Src: 192.168.50.13 (192.168.50.13), Dst: 207.46.113.50 (207.46.113.50)
    Version: 4
    Header Length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 460
    Identification: 0x3e6c (15980)
  ⊞ Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 128
    Protocol: TCP (6)
  ⊞ Header checksum: 0x87a9 [validation disabled]
    Source: 192.168.50.13 (192.168.50.13)
    Destination: 207.46.113.50 (207.46.113.50)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
⊞ Transmission Control Protocol, Src Port: 49402 (49402), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 420
⊞ Hypertext Transfer Protocol

0030  40 29 77 c6 00 00 47 45  54 20 2f 73 63 72 69 70   @)w...GE T /scrip
0040  74 73 2f 34 2e 32 2f 77  6f 6c 2f 6d 6f 64 65 72   ts/4.2/w ol/moder
0050  6e 69 7a 72 2e 77 6f 6c  2e 6a 73 20 48 54 54 50   nizr.wol .js HTTP
0060  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 77 69 6e 64   /1.1..Ho st: wind
0070  6f 77 73 2e 6d 69 63 72  6f 73 6f 66 74 2e 63 6f   ows.micr osoft.co

Hypertext Transfer Protocol (http), 420 bytes   Packets: 31049 · Displayed: 102 (0.3%) · Load time: 0:00.312   Profile: ...
```

6) Screenshot showing source and destination port of requested packet. _


logs.pcapng [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save
Profile: ...
3:36 PM
8/19/2014

Filter: http.request.method==GET

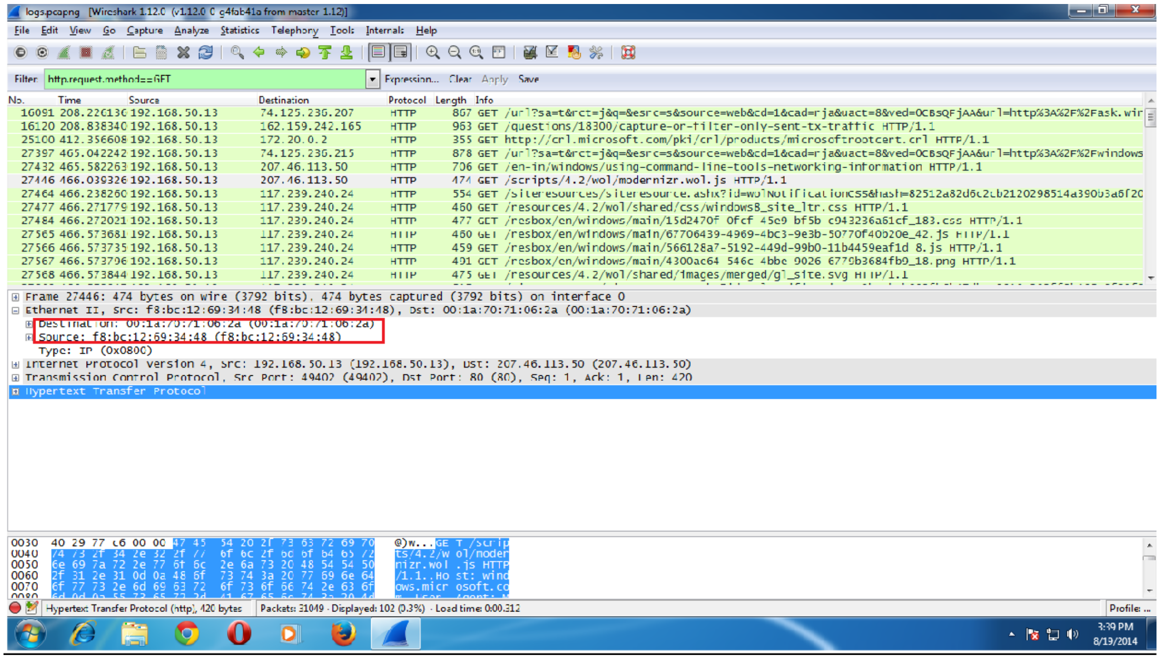
```
No.   Time       Source          Destination      Protocol Length Info
16091 208.226136 192.168.50.13   74.125.236.207   HTTP     867 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fask.wir
16120 208.838340 192.168.50.13   162.159.242.165  HTTP     963 GET /questions/18300/capture-or-filter-only-sent-tx-traffic HTTP/1.1
25100 412.356608 192.168.50.13   172.20.0.2       HTTP     355 GET http://crl.microsoft.com/pki/crl/products/microsoftrootcert.crl HTTP/1.1
27397 465.042242 192.168.50.13   74.125.236.215   HTTP     878 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fwindows
27432 465.582263 192.168.50.13   207.46.113.50    HTTP     706 GET /en-in/windows/using-command-line-tools-networking-information HTTP/1.1
27446 466.039326 192.168.50.13   207.46.113.50    HTTP     474 GET /scripts/4.2/wol/modernizr.wol.js HTTP/1.1
27464 466.238260 192.168.50.13   117.239.240.24   HTTP     554 GET /siteresources/siteresource.ashx?id=wolNotificationCSS&hash=82512a82d6c2cb2120298514a390b3a6f20
27477 466.271779 192.168.50.13   117.239.240.24   HTTP     460 GET /resources/4.2/wol/shared/css/windows8_site_ltr.css HTTP/1.1
27484 466.272021 192.168.50.13   117.239.240.24   HTTP     477 GET /resbox/en/windows/main/15d2470f-0fcf-45e9-bf5b-c943236a61cf_183.css HTTP/1.1
27565 466.573681 192.168.50.13   117.239.240.24   HTTP     460 GET /resbox/en/windows/main/67706439-4969-4bc3-9e3b-50770f40b20e_42.js HTTP/1.1
27566 466.573735 192.168.50.13   117.239.240.24   HTTP     459 GET /resbox/en/windows/main/566128a7-5192-449d-99b0-11b4459eaf1d_8.js HTTP/1.1
27567 466.573796 192.168.50.13   117.239.240.24   HTTP     491 GET /resbox/en/windows/main/4300ae64-546c-4bbe-9026-6779b3684fb9_18.png HTTP/1.1
27568 466.573844 192.168.50.13   117.239.240.24   HTTP     475 GET /resources/4.2/wol/shared/images/merged/gl_site.svg HTTP/1.1
```

```
⊞ Frame 27446: 474 bytes on wire (3792 bits), 474 bytes captured (3792 bits) on interface 0
⊞ Ethernet II, Src: f8:bc:12:69:34:48 (f8:bc:12:69:34:48), Dst: 00:1a:70:71:06:2a (00:1a:70:71:06:2a)
⊞ Internet Protocol Version 4, Src: 192.168.50.13 (192.168.50.13), Dst: 207.46.113.50 (207.46.113.50)
⊟ Transmission Control Protocol, Src Port: 49402 (49402), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 420
    Source Port: 49402 (49402)
    Destination Port: 80 (80)
    [Stream index: 186]
    [TCP Segment Len: 420]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 421    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Header Length: 20 bytes
  ⊞ .... 0000 0001 1000 = Flags: 0x018 (PSH, ACK)
    Window size value: 16425
    [Calculated window size: 65700]
    [Window size scaling factor: 4]
  ⊞ Checksum: 0x77c6 [validation disabled]
    Urgent pointer: 0
```

```
0030  40 29 77 c6 00 00 47 45  54 20 2f 73 63 72 69 70   @)w...GE T /scrip
0040  74 73 2f 34 2e 32 2f 77  6f 6c 2f 6d 6f 64 65 72   ts/4.2/w ol/moder
0050  6e 69 7a 72 2e 77 6f 6c  2e 6a 73 20 48 54 54 50   nizr.wol .js HTTP
0060  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 77 69 6e 64   /1.1..Ho st: wind
0070  6f 77 73 2e 6d 69 63 72  6f 73 6f 66 74 2e 63 6f   ows.micr osoft.co
```

Hypertext Transfer Protocol (http), 420 bytes | Packets: 31049 · Displayed: 102 (0.3%) · Load time: 0:00.312

7) Screenshot of getting MAC address of source and destination system


logs.pcapng [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save
Profile: ...
3:39 PM
8/19/2014

Filter: http.request.method==GET

```
No.   Time       Source          Destination      Protocol Length Info
16091 208.226136 192.168.50.13   74.125.236.207   HTTP     867 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fask.wir
16120 208.838340 192.168.50.13   162.159.242.165  HTTP     963 GET /questions/18300/capture-or-filter-only-sent-tx-traffic HTTP/1.1
25100 412.356608 192.168.50.13   172.20.0.2       HTTP     355 GET http://crl.microsoft.com/pki/crl/products/microsoftrootcert.crl HTTP/1.1
27397 465.042242 192.168.50.13   74.125.236.215   HTTP     878 GET /url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBsQFjAA&url=http%3A%2F%2Fwindows
27432 465.582263 192.168.50.13   207.46.113.50    HTTP     706 GET /en-in/windows/using-command-line-tools-networking-information HTTP/1.1
27446 466.039326 192.168.50.13   207.46.113.50    HTTP     474 GET /scripts/4.2/wol/modernizr.wol.js HTTP/1.1
27464 466.238260 192.168.50.13   117.239.240.24   HTTP     554 GET /siteresources/siteresource.ashx?id=wolNotificationCSS&hash=82512a82d6c2cb2120298514a390b3a6f20
27477 466.271779 192.168.50.13   117.239.240.24   HTTP     460 GET /resources/4.2/wol/shared/css/windows8_site_ltr.css HTTP/1.1
27484 466.272021 192.168.50.13   117.239.240.24   HTTP     477 GET /resbox/en/windows/main/15d2470f-0fcf-45e9-bf5b-c943236a61cf_183.css HTTP/1.1
27565 466.573681 192.168.50.13   117.239.240.24   HTTP     460 GET /resbox/en/windows/main/67706439-4969-4bc3-9e3b-50770f40b20e_42.js HTTP/1.1
27566 466.573735 192.168.50.13   117.239.240.24   HTTP     459 GET /resbox/en/windows/main/566128a7-5192-449d-99b0-11b4459eaf1d_8.js HTTP/1.1
27567 466.573796 192.168.50.13   117.239.240.24   HTTP     491 GET /resbox/en/windows/main/4300ae64-546c-4bbe-9026-6779b3684fb9_18.png HTTP/1.1
27568 466.573844 192.168.50.13   117.239.240.24   HTTP     475 GET /resources/4.2/wol/shared/images/merged/gl_site.svg HTTP/1.1
```

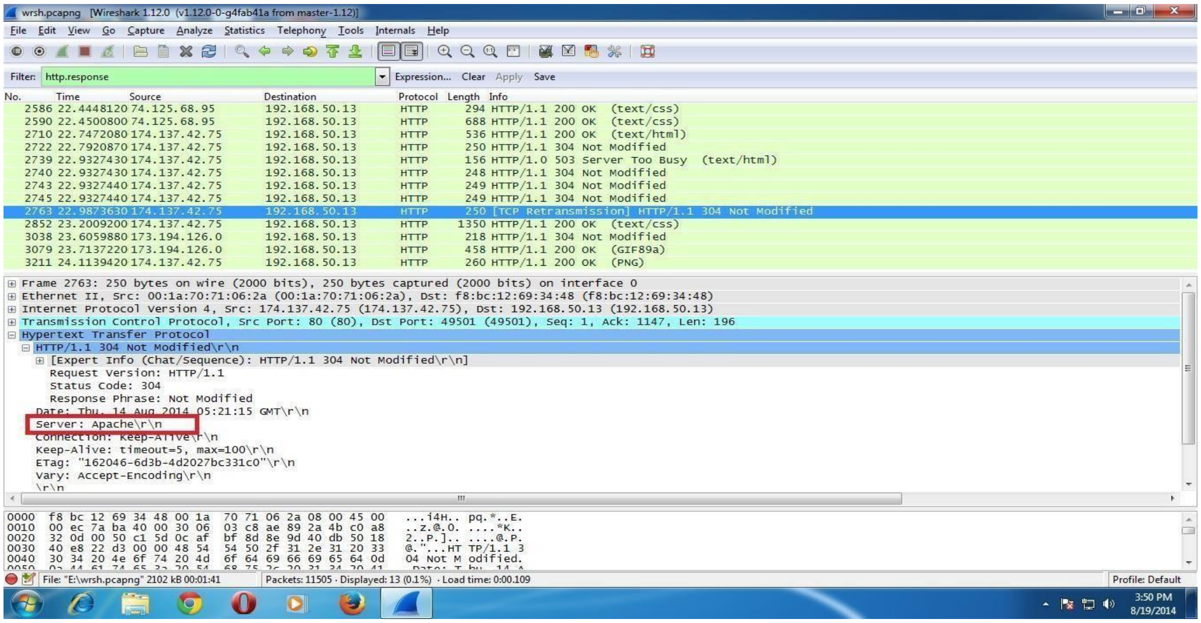
```
⊞ Frame 27446: 474 bytes on wire (3792 bits), 474 bytes captured (3792 bits) on interface 0
⊟ Ethernet II, Src: f8:bc:12:69:34:48 (f8:bc:12:69:34:48), Dst: 00:1a:70:71:06:2a (00:1a:70:71:06:2a)
  ⊞ Destination: 00:1a:70:71:06:2a (00:1a:70:71:06:2a)
  ⊞ Source: f8:bc:12:69:34:48 (f8:bc:12:69:34:48)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 192.168.50.13 (192.168.50.13), Dst: 207.46.113.50 (207.46.113.50)
⊞ Transmission Control Protocol, Src Port: 49402 (49402), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 420
⊞ Hypertext Transfer Protocol
```

```
0030  40 29 77 c6 00 00 47 45  54 20 2f 73 63 72 69 70   @)w...GE T /scrip
0040  74 73 2f 34 2e 32 2f 77  6f 6c 2f 6d 6f 64 65 72   ts/4.2/w ol/moder
0050  6e 69 7a 72 2e 77 6f 6c  2e 6a 73 20 48 54 54 50   nizr.wol .js HTTP
0060  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 77 69 6e 64   /1.1..Ho st: wind
0070  6f 77 73 2e 6d 69 63 72  6f 73 6f 66 74 2e 63 6f   ows.micr osoft.co
```

Hypertext Transfer Protocol (http), 420 bytes | Packets: 31049 · Displayed: 102 (0.3%) · Load time: 0:00.312

8) Screenshot for getting server information from response packets

Type "http.response" in the filter bar to get response packets.



wrsh.pcapng [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save
3:50 PM
8/19/2014
Profile: Default

Filter: http.response

```
No.   Time       Source          Destination    Protocol Length Info
 2586 22.4448120 74.125.68.95    192.168.50.13  HTTP      294 HTTP/1.1 200 OK  (text/css)
 2590 22.4500800 74.125.68.95    192.168.50.13  HTTP      688 HTTP/1.1 200 OK  (text/css)
 2710 22.7472080 174.137.42.75   192.168.50.13  HTTP      536 HTTP/1.1 200 OK  (text/html)
 2722 22.7920870 174.137.42.75   192.168.50.13  HTTP      250 HTTP/1.1 304 Not Modified
 2739 22.9327430 174.137.42.75   192.168.50.13  HTTP      156 HTTP/1.0 503 Server Too Busy  (text/html)
 2740 22.9327430 174.137.42.75   192.168.50.13  HTTP      248 HTTP/1.1 304 Not Modified
 2743 22.9327440 174.137.42.75   192.168.50.13  HTTP      249 HTTP/1.1 304 Not Modified
 2745 22.9327440 174.137.42.75   192.168.50.13  HTTP      249 HTTP/1.1 304 Not Modified
 2763 22.9873630 174.137.42.75   192.168.50.13  HTTP      250 [TCP Retransmission] HTTP/1.1 304 Not Modified
 2852 23.2009200 174.137.42.75   192.168.50.13  HTTP     1350 HTTP/1.1 200 OK  (text/css)
 3038 23.6059880 173.194.126.0   192.168.50.13  HTTP      218 HTTP/1.1 304 Not Modified
 3079 23.7137220 173.194.126.0   192.168.50.13  HTTP      458 HTTP/1.1 200 OK  (GIF89a)
 3211 24.1139420 174.137.42.75   192.168.50.13  HTTP      260 HTTP/1.1 200 OK  (PNG)
```

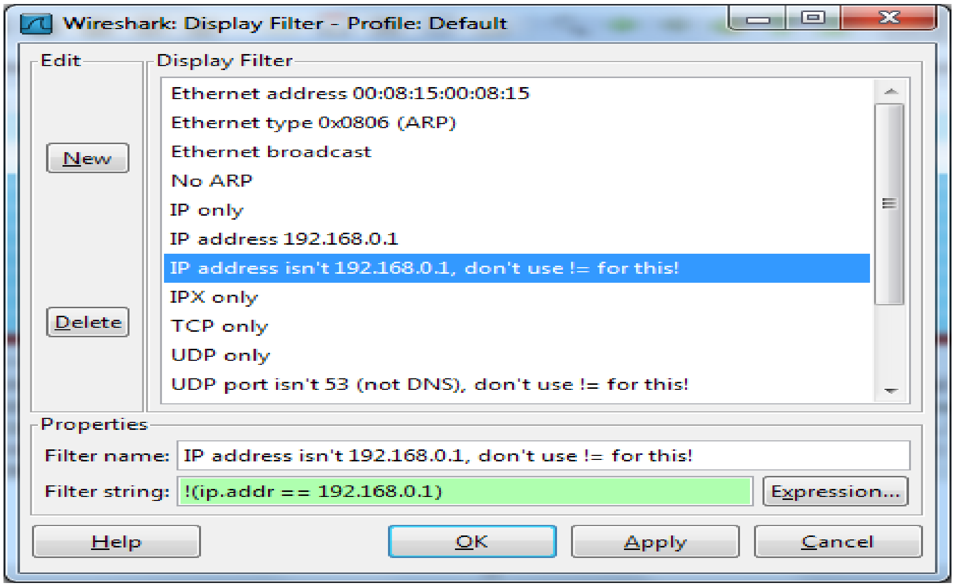
```
⊞ Frame 2763: 250 bytes on wire (2000 bits), 250 bytes captured (2000 bits) on interface 0
⊞ Ethernet II, Src: 00:1a:70:71:06:2a (00:1a:70:71:06:2a), Dst: f8:bc:12:69:34:48 (f8:bc:12:69:34:48)
⊞ Internet Protocol Version 4, Src: 174.137.42.75 (174.137.42.75), Dst: 192.168.50.13 (192.168.50.13)
⊞ Transmission Control Protocol, Src Port: 80 (80), Dst Port: 49501 (49501), Seq: 1, Ack: 1147, Len: 196
⊟ Hypertext Transfer Protocol
  ⊟ HTTP/1.1 304 Not Modified\r\n
    ⊞ [Expert Info (Chat/Sequence): HTTP/1.1 304 Not Modified\r\n]
      Request Version: HTTP/1.1
      Status Code: 304
      Response Phrase: Not Modified
    Date: Thu, 14 Aug 2014 05:21:15 GMT\r\n
    Server: Apache\r\n
    Connection: Keep-Alive\r\n
    Keep-Alive: timeout=5, max=100\r\n
    ETag: "162046-6d3b-4d2027bc331c0"\r\n
    Vary: Accept-Encoding\r\n
    \r\n
```

```
0000  f8 bc 12 69 34 48 00 1a  70 71 06 2a 08 00 45 00   ...i4H.. pq.*..E.
0010  00 ec 7a ba 40 00 30 06  03 c8 ae 89 2a 4b c0 a8   ..z.@.0. ....*K..
0020  32 0d 00 50 c1 5d 0c af  bf 8d 8e 9d 40 db 50 18   2..P.].. ....@.P.
0030  40 e8 22 d3 00 00 48 54  54 50 2f 31 2e 31 20 33   @."...HT TP/1.1 3
0040  30 34 20 4e 6f 74 20 4d  6f 64 69 66 69 65 64 0d   04 Not M odified.
```

File: "E:\wrsh.pcapng" 2102 kB 00:01:41 | Packets: 11505 · Displayed: 13 (0.1%) · Load time: 0:00.109

Wireshark Filtering Methods

To create a new filter: Click Analyze menu select Display Filters

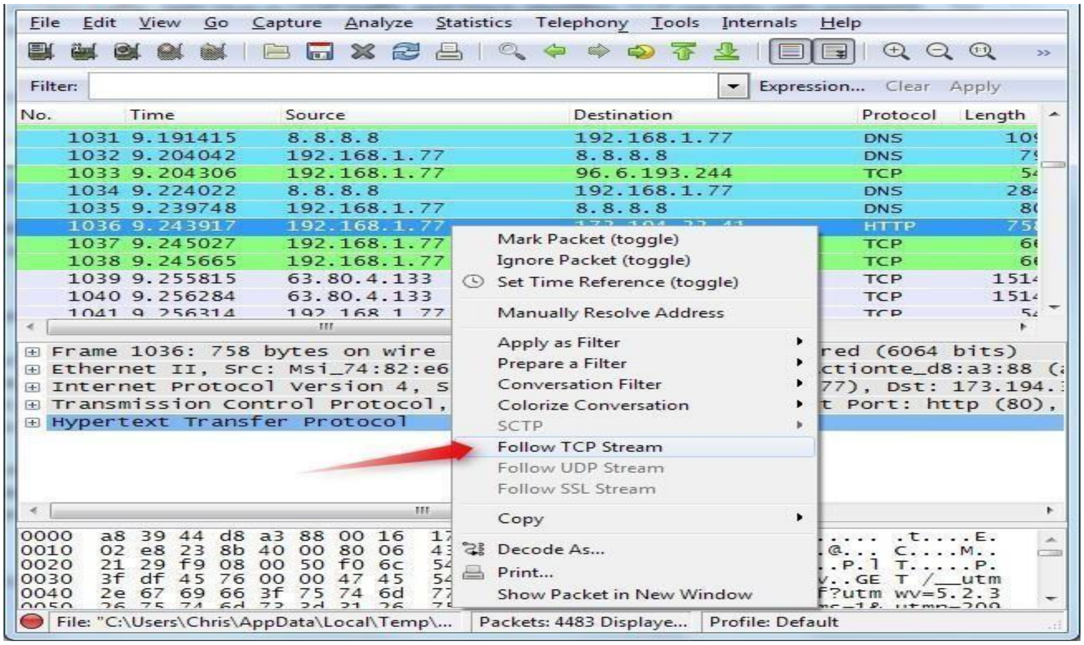


Ethernet address 00:08:15:00:08:15
Ethernet type 0x0806 (ARP)
Ethernet broadcast
No ARP
IP only
IP address 192.168.0.1
IP address isn't 192.168.0.1, don't use != for this!
IPX only
TCP only
UDP only
UDP port isn't 53 (not DNS), don't use != for this!

Filter name: IP address isn't 192.168.0.1, don't use != for this!

Filter string: !(ip.addr == 192.168.0.1)

Right click on a packet and select Follow TCP Stream



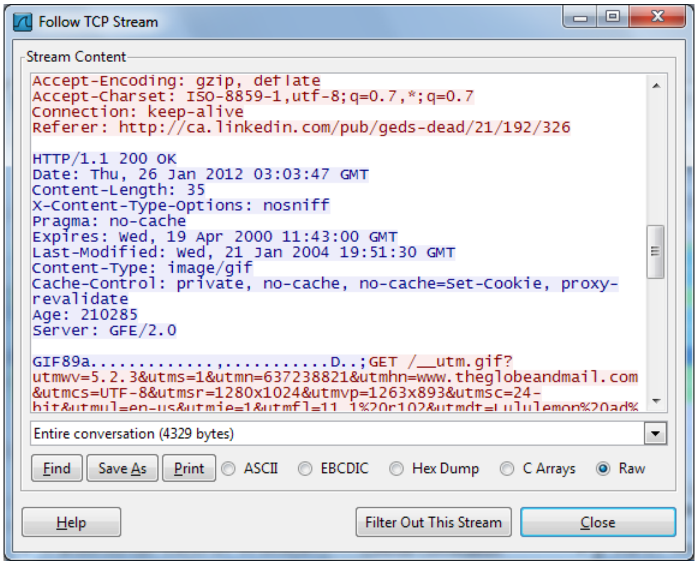
```
No.      Time        Source          Destination       Protocol  Length
   1031 9.191415     8.8.8.8         192.168.1.77      DNS        10
   1032 9.204042     192.168.1.77    8.8.8.8           DNS        7
   1033 9.204306     192.168.1.77    96.6.193.244      TCP        5
   1034 9.224022     8.8.8.8         192.168.1.77      DNS        28
   1035 9.239748     192.168.1.77    8.8.8.8           DNS        8
   1036 9.243917     192.168.1.77                      HTTP       75
   1037 9.245027     192.168.1.77                      TCP        6
   1038 9.245665     192.168.1.77                      TCP        6
   1039 9.255815     63.80.4.133                       TCP        151
   1040 9.256284     63.80.4.133                       TCP        151
   1041 9.256314     192.168.1.77                      TCP        5
```

```
⊞ Frame 1036: 758 bytes on wire                red (6064 bits)
⊞ Ethernet II, Src: Msi_74:82:e6              ctionte_d8:a3:88 (
⊞ Internet Protocol Version 4, S              77), Dst: 173.194.
⊞ Transmission Control Protocol,              t Port: http (80),
⊞ Hypertext Transfer Protocol
```

```
0000  a8 39 44 d8 a3 88 00 16  17            .....  .t....E.
0010  02 e8 23 8b 40 00 80 06  43            .@...  C....M..
0020  21 29 f9 08 00 50 f0 6c  54            ..P.l  T.....P.
0030  3f df 45 76 00 00 47 45  54            v..GE  T /__utm
0040  2e 67 69 66 3f 75 74 6d  77            f?utm  wv=5.2.3
```

File: "C:\Users\Chris\AppData\Local\Temp\... Packets: 4483 Displaye... Profile: Default

Full conversation between the client and the server

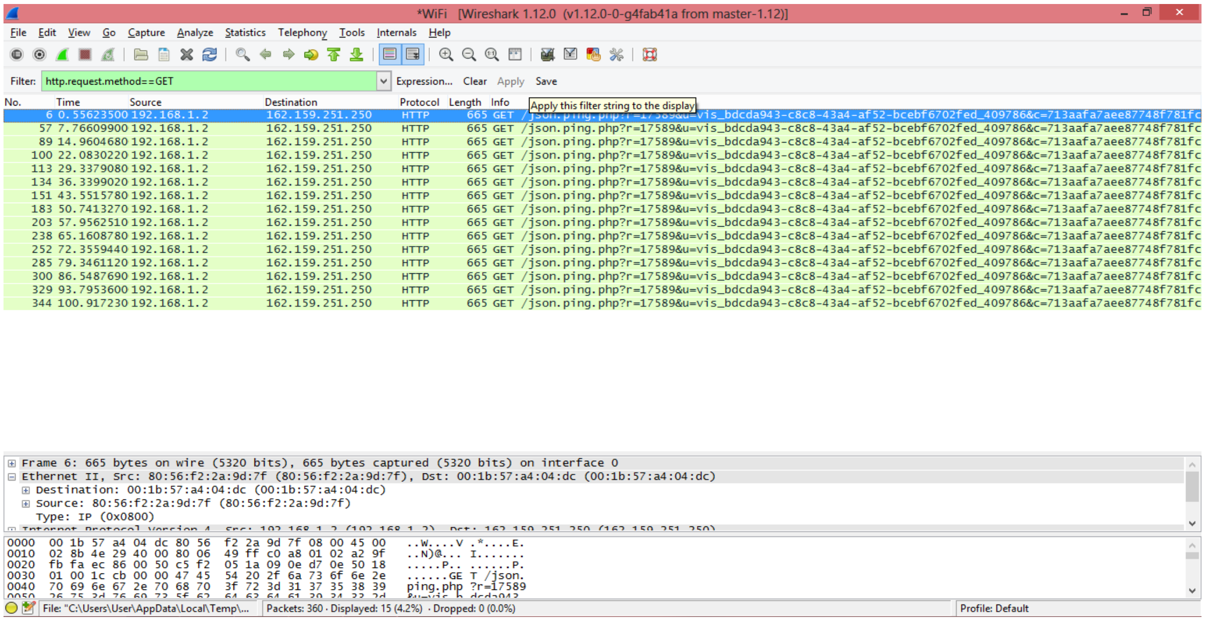


```
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: keep-alive
Referer: http://ca.linkedin.com/pub/geds-dead/21/192/326

HTTP/1.1 200 OK
Date: Thu, 26 Jan 2012 03:03:47 GMT
Content-Length: 35
X-Content-Type-Options: nosniff
Pragma: no-cache
Expires: Wed, 19 Apr 2000 11:43:00 GMT
Last-Modified: Wed, 21 Jan 2004 19:51:30 GMT
Content-Type: image/gif
Cache-Control: private, no-cache, no-cache=Set-Cookie, proxy-
revalidate
Age: 210285
Server: GFE/2.0

GIF89a...............,...........D..;GET /__utm.gif?
utmwv=5.2.3&utms=1&utmn=637238821&utmhn=www.theglobeandmail.com
&utmcs=UTF-8&utmsr=1280x1024&utmvp=1263x893&utmsc=24-
bit&utmul=en-us&utmje=1&utmfl=11.1%20r102&utmdt=Lululemon%20ad%
```

Entire conversation (4329 bytes)

a) Count total number of HTTP GET requests

Command for this is http.request.method==GET



*WiFi [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save
Apply this filter string to the display

Filter: http.request.method==GET

```
No.   Time       Source        Destination      Protocol Length Info
    6 0.55623500 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
   57 7.76609900 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
   89 14.9604680 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  100 22.0830220 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  113 29.3379080 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  134 36.3399020 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  151 43.5515780 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  183 50.7413270 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  203 57.9562510 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  238 65.1608780 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  252 72.3559440 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  285 79.3461120 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  300 86.5487690 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  329 93.7953600 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
  344 100.917230 192.168.1.2   162.159.251.250  HTTP     665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781fc
```

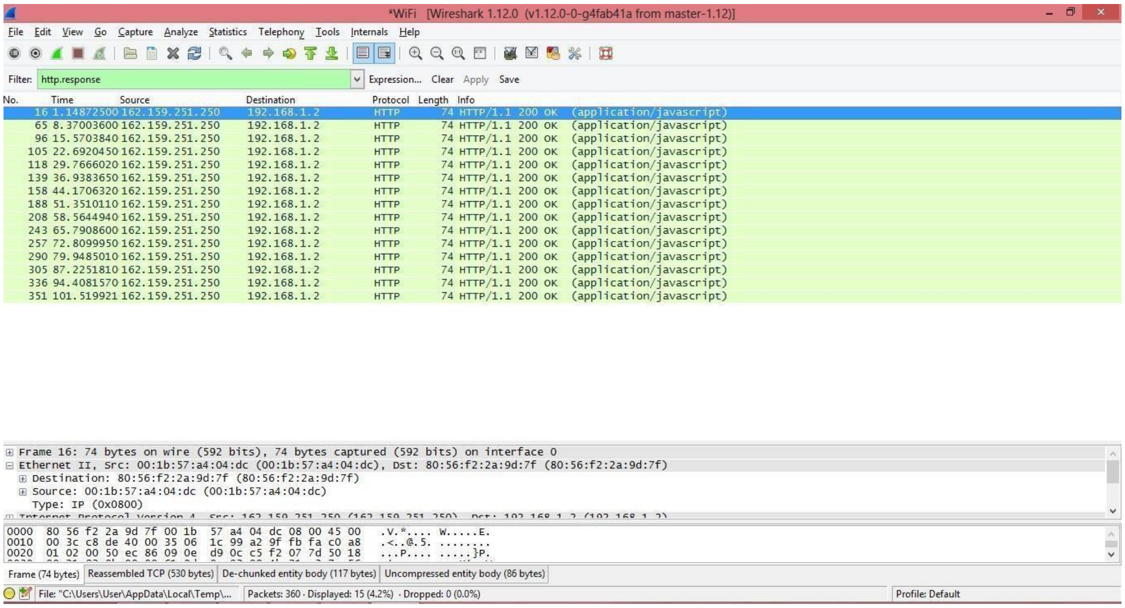
```
⊞ Frame 6: 665 bytes on wire (5320 bits), 665 bytes captured (5320 bits) on interface 0
⊟ Ethernet II, Src: 80:56:f2:2a:9d:7f (80:56:f2:2a:9d:7f), Dst: 00:1b:57:a4:04:dc (00:1b:57:a4:04:dc)
  ⊞ Destination: 00:1b:57:a4:04:dc (00:1b:57:a4:04:dc)
  ⊞ Source: 80:56:f2:2a:9d:7f (80:56:f2:2a:9d:7f)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 192.168.1.2 (192.168.1.2), Dst: 162.159.251.250 (162.159.251.250)
```

```
0000  00 1b 57 a4 04 dc 80 56  f2 2a 9d 7f 08 00 45 00   ..W....V .*....E.
0010  02 8b 4e 29 40 00 80 06  49 ff c0 a8 01 02 a2 9f   ..N)@... I.......
0020  fb fa ec 86 00 50 c5 f2  05 1a 09 0e d7 0e 50 18   .....P.. ......P.
0030  01 00 1c cb 00 00 47 45  54 20 2f 6a 73 6f 6e 2e   ......GE T /json.
0040  70 69 6e 67 2e 70 68 70  3f 72 3d 31 37 35 38 39   ping.php ?r=17589
0050  26 75 3d 76 69 73 5f 62  64 63 64 61 39 34 33 2d   &u=vis_b dcda943-
```

File: "C:\Users\User\AppData\Local\Temp\... | Packets: 360 · Displayed: 15 (4.2%) · Dropped: 0 (0.0%) | Profile: Default

b) Identify sever software running on server side

Command used is http.response then check the http part of the first response.



*WiFi [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

Filter: http.response

```
No.   Time       Source           Destination  Protocol Length Info
   16 1.14872500 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
   65 8.37003600 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
   96 15.5703840 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  105 22.6920450 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  118 29.7666020 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  139 36.9383650 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  158 44.1706320 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  188 51.3510110 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  208 58.5644940 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  243 65.7908600 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  257 72.8099950 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  290 79.9485010 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  305 87.2251810 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  336 94.4081570 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
  351 101.519921 162.159.251.250  192.168.1.2  HTTP     74 HTTP/1.1 200 OK  (application/javascript)
```

```
⊞ Frame 16: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
⊟ Ethernet II, Src: 00:1b:57:a4:04:dc (00:1b:57:a4:04:dc), Dst: 80:56:f2:2a:9d:7f (80:56:f2:2a:9d:7f)
  ⊞ Destination: 80:56:f2:2a:9d:7f (80:56:f2:2a:9d:7f)
  ⊞ Source: 00:1b:57:a4:04:dc (00:1b:57:a4:04:dc)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 162.159.251.250 (162.159.251.250), Dst: 192.168.1.2 (192.168.1.2)
```

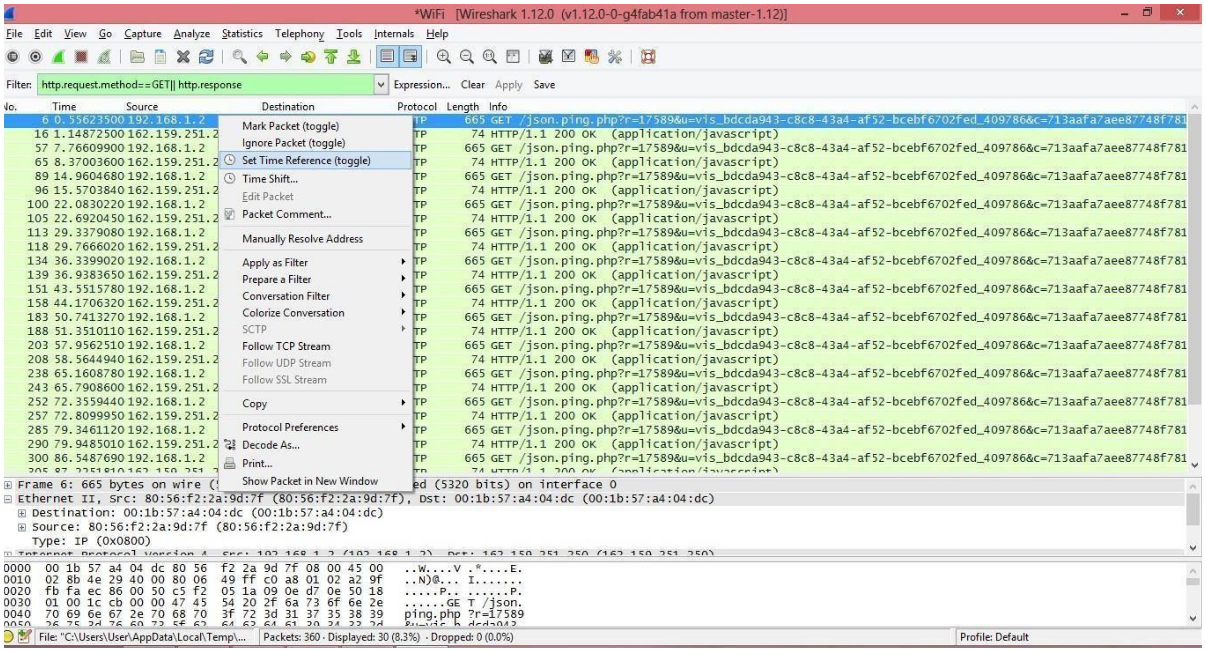
```
0000  80 56 f2 2a 9d 7f 00 1b  57 a4 04 dc 08 00 45 00   .V.*.... W.....E.
0010  00 3c c8 de 40 00 35 06  1c 99 a2 9f fb fa c0 a8   .<..@.5. ........
0020  01 02 00 50 ec 86 09 0e  d9 0c c5 f2 07 7d 50 18   ...P.... .....}P.
```

Frame (74 bytes) | Reassembled TCP (530 bytes) | De-chunked entity body (117 bytes) | Uncompressed entity body (86 bytes)

File: "C:\Users\User\AppData\Local\Temp\... | Packets: 360 · Displayed: 15 (4.2%) · Dropped: 0 (0.0%) | Profile: Default

4) Identify time elapsed between first http GET request and http response.

- Command used is http.request.method==GET|| http.response
- Identify the first request and response packet.
- Right click on first request and set it as time reference.
- Identify the time elapsed for the response packet.



Filter: http.request.method==GET|| http.response

```
No.   Time        Source          Destination   Protocol Length Info
    6 0.55623500  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
   16 1.14872500  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
   57 7.76609900  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
   65 8.37003600  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
   89 14.9604680  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
   96 15.5703840  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  100 22.0830220  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  105 22.6920450  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  113 29.3379080  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  118 29.7666020  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  134 36.3399020  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  139 36.9383650  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  151 43.5515780  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  158 44.1706320  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  183 50.7413270  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  188 51.3510110  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  203 57.9562510  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  208 58.5644940  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  238 65.1608780  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  243 65.7908600  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  252 72.3559440  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  257 72.8099950  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  285 79.3461120  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  290 79.9485010  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
  300 86.5487690  192.168.1.2                   TP    665 GET /json.ping.php?r=17589&u=vis_bdcda943-c8c8-43a4-af52-bcebf6702fed_409786&c=713aafa7aee87748f781
  305 87.2251810  162.159.251.2                 TP     74 HTTP/1.1 200 OK  (application/javascript)
```

```
⊞ Frame 6: 665 bytes on wire (                          ed (5320 bits) on interface 0
⊟ Ethernet II, Src: 80:56:f2:2a:9d:7f (80:56:f2:2a:9d:7f), Dst: 00:1b:57:a4:04:dc (00:1b:57:a4:04:dc)
  ⊞ Destination: 00:1b:57:a4:04:dc (00:1b:57:a4:04:dc)
  ⊞ Source: 80:56:f2:2a:9d:7f (80:56:f2:2a:9d:7f)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 192.168.1.2 (192.168.1.2), Dst: 162.159.251.250 (162.159.251.250)
```

```
0000  00 1b 57 a4 04 dc 80 56  f2 2a 9d 7f 08 00 45 00   ..W....V .*....E.
0010  02 8b 4e 29 40 00 80 06  49 ff c0 a8 01 02 a2 9f   ..N)@... I.......
0020  fb fa ec 86 00 50 c5 f2  05 1a 09 0e d7 0e 50 18   .....P.. ......P.
0030  01 00 1c cb 00 00 47 45  54 20 2f 6a 73 6f 6e 2e   ......GE T /json.
0040  70 69 6e 67 2e 70 68 70  3f 72 3d 31 37 35 38 39   ping.php ?r=17589
0050  26 75 3d 76 69 73 5f 62  64 63 64 61 39 34 33 2d   &u=vis_b dcda943-
```

File: "C:\Users\User\AppData\Local\Temp\... Packets: 360 · Displayed: 30 (8.3%) · Dropped: 0 (0.0%)

Result:

Familiarized and successfully analyzed packets using wireshark and performed filtering methods.

Experiment No:-04 Date: …/... /2024

Vulnerability testing tool

Aim:

To familiarize with the use of NMAP for network exploration and security auditing.

Theoretical Background:-

Nmap (Network Mapper) is a free, open-source port scanner available for both UNIX and Windows. It‘s very useful for network exploration and security auditing. It was designed to rapidly scan large networks, although it works fine against single hosts. Main N-MAP features are:-

1. Ping Sweeping :- Identifying computers on a network.

2. Port Scanning :- Enumerating the open Ports on one or more target computers.

3. OS Detection :- Remotely determining the operating system and some hardware characteristics of network devices.

The six port states recognized by Nmap are:

1. open :- An application is actively accepting TCP connections, UDP datagrams or SCTP associations on this port. .

2. closed :- A closed port is accessible (it receives and responds to Nmap probe packets), but there is no application listening on it. They can be helpful in showing that a host is up on an IP address (host discovery, or ping scanning), and as part of OS detection.

3. filtered :- Nmap cannot determine whether the port is open because packet filtering prevents its probes from reaching the port. The filtering could be from a dedicated firewall device, router rules, or host-based firewall software. information.

4. unfiltered :- The unfiltered state means that a port is accessible, but Nmap is unable to determine whether it is open or closed. Only the ACK scan, which is used to map firewall rulesets, classifies ports into this state.

5. open|filtered :- Nmap places ports in this state when it is unable to determine whether a port is open or filtered. This occurs for scan types in which open ports give no response. T

6. closed|filtered :- This state is used when Nmap is unable to determine whether a port is closed or filtered. It is only used for the IP ID idle scan.

Most of the scan types are only available to privileged users. This is because they send and receive raw packets, which requires root access on linux.Some of the scanning techiniques are listed below.

b) sT(TCP connect() scan)

These scans are so called because UNIX sockets programming uses a system call named connect() to begin a TCP connection to a remote site. If connect() succeeds, a connection was made. If it fails, the connection could not be made. Once the scan is completed, ports to which a connection could be established are listed as open, the rest are said to be closed.

c) sS(SYN Stealth scan)

SYN or Stealth scanning makes use of this procedure by sending a SYN packet and looking at the response. If SYN/ACK is sent back, the port is open and the remote end is trying to open a TCP connection.

d) sU (UDP scan) UDP scan is activated with the -sU option. UDP scan works by sending a UDP packet to every targeted port.

e) sY (SCTP INIT scan)

SCTP is a relatively new alternative to the TCP and UDP protocols, combining most characteristics of TCP and UDP, and also adding new features like multi-homing and multi-streaming. SCTP INIT scan is the SCTP equivalent of a TCP SYN scan. It also allows clear, reliable differentiation between the open, closed, and filtered states. e)sN; -sF; -sX (TCP NULL, FIN, and Xmas scans)

These three scan types exploit a subtle loophole in the TCP RFC to differentiate between open and closed ports. Null scan (-sN):-Does not set any bits (TCP flag header is 0)

FIN scan (-sF):Sets just the TCP FIN bit.

Xmas scan (-sX):-Sets the FIN, PSH, and URG flags, lighting the packet up like a Christmas tree

f)-sA (TCP ACK scan) This scan is different than the others discussed so far in that it never determines open (or even open|filtered) ports. It is used to map out firewall rulesets, determining whether they are stateful or not and which ports are filtered. g)-sW (TCP Window scan)

Window scan is exactly the same as ACK scan except that it exploits an implementation detail of certain systems to differentiate open ports from closed ones, rather than always printing unfiltered when a RST is returned. It does this by examining the TCP Window field of the RST packets returned. h)-sM (TCP Maimon scan)

The Maimon scan is named after its discoverer, Uriel Maimon. This technique is exactly the same as NULL, FIN, and Xmas scans, except that the probe is FIN/ACK. i)-sO (IP protocol scan)

IP protocol scan allows you to determine which IP protocols (TCP, ICMP, IGMP, etc.) are supported by target machines. j)-b <FTP relay host>(FTP bounce scan)

FTP protocol allows a user to connect to one FTP server, then ask that files be sent to a third-party server. Such a feature is ripe for abuse on many levels, so most servers have ceased supporting it. One of the abuses this feature allows is causing the FTP server to port scan other hosts. Simply ask the FTP server to send a file to each interesting port of a target host in turn. The error message will describe whether the port is open or not. Nmap supports FTP bounce scan with the -b option. k) -O (Operating System Detection) Allows the user to use TCP/IP fingerprinting to determine the operating system of the remote host.

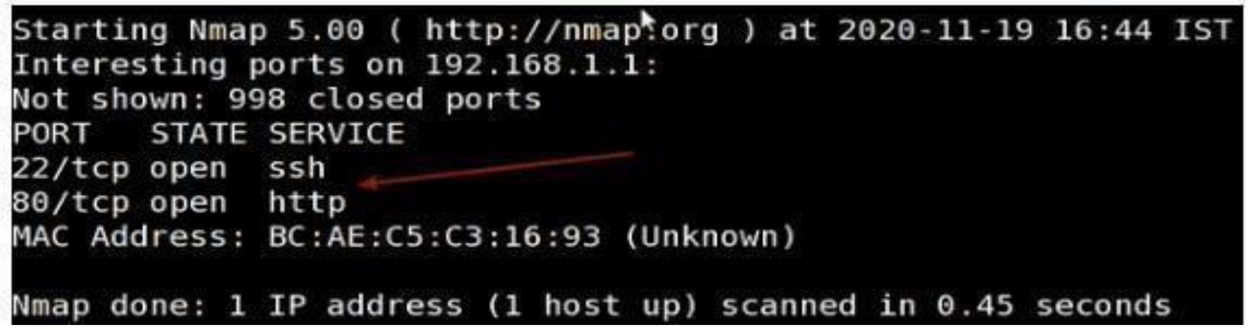
Commands and sample outputs

1: Scan a single host or an IP address (IPv4)

nmap server1.cyberciti.biz

nmap -v server1.cyberciti.biz

nmap 192.168.1.1 [Sample output]



```
Starting Nmap 5.00 ( http://nmap.org ) at 2020-11-19 16:44 IST
Interesting ports on 192.168.1.1:
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: BC:AE:C5:C3:16:93 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.45 seconds
```

2: Scan multiple IP address or subnet (IPv4)

nmap 192.168.1.1 192.168.1.2 192.168.1.3

nmap 192.168.1.1,2,3

Scan a range of IP address too: **nmap 192.168.1.1-20**

Scan a range of IP address using a wildcard: **nmap 192.168.1.***

Scan an entire subnet: **nmap 192.168.1.0/24**

3: Read list of hosts/networks from a file (IPv4)

The -iL option allows you to read the list of target systems using a text file. This is useful to scan a large number of hosts/networks. Create a text file as follows:

```
cat > /tmp/test.txt
```

The syntax is: nmap -iL /tmp/test.txt

4: Excluding hosts/networks (IPv4)

When scanning a large number of hosts/networks you can exclude hosts from a scan:

```
nmap 192.168.1.0/24 --exclude 192.168.1.5
```

```
nmap 192.168.1.0/24 --exclude 192.168.1.5,192.168.1.254
```

OR

exclude list from a file called **/tmp/exclude.txt**

```
nmap -iL /tmp/scanlist.txt –exclude file/tmp/exclude.txt
```

5: Turn on OS and version detection scanning script (IPv4) nmap

```
-A 192.168.1.254
```

```
nmap -v -A 192.168.1.1
```

```
nmap -A -iL /tmp/scanlist.txt
```

6: Find out if a host/network is protected by a firewall

```
nmap -sA 192.168.1.254
```

```
nmap -sA server1.cyberciti.biz
```

7: Scan a host when protected by the firewall

```
nmap -PN 192.168.1.1

nmap -PN server1.cyberciti.biz
```

8: Scan an IPv6 host/address

The -6 option enable IPv6 scanning. The syntax is:

```
nmap -6 IPv6-Address-Here

nmap -6 server1.cyberciti.biz

nmap -6 2607:f0d0:1002:51::4

nmap -v A -6 2607:f0d0:1002:51::4
```

9: Scan a network and find out which servers and devices are up and running

This is known as host discovery or ping scan:

```
nmap -sP 192.168.1.0/24
```

Sample outputs:

```
Host 192.168.1.1 is up (0.00035s latency).

MAC Address: BC:AE:C5:C3:16:93 (Unknown)

Host 192.168.1.2 is up (0.0038s latency).

MAC Address: 74:44:01:40:57:FB (Unknown)

Host 192.168.1.5 is up.
```

Host nas03 (192.168.1.12) is up (0.0091s latency).

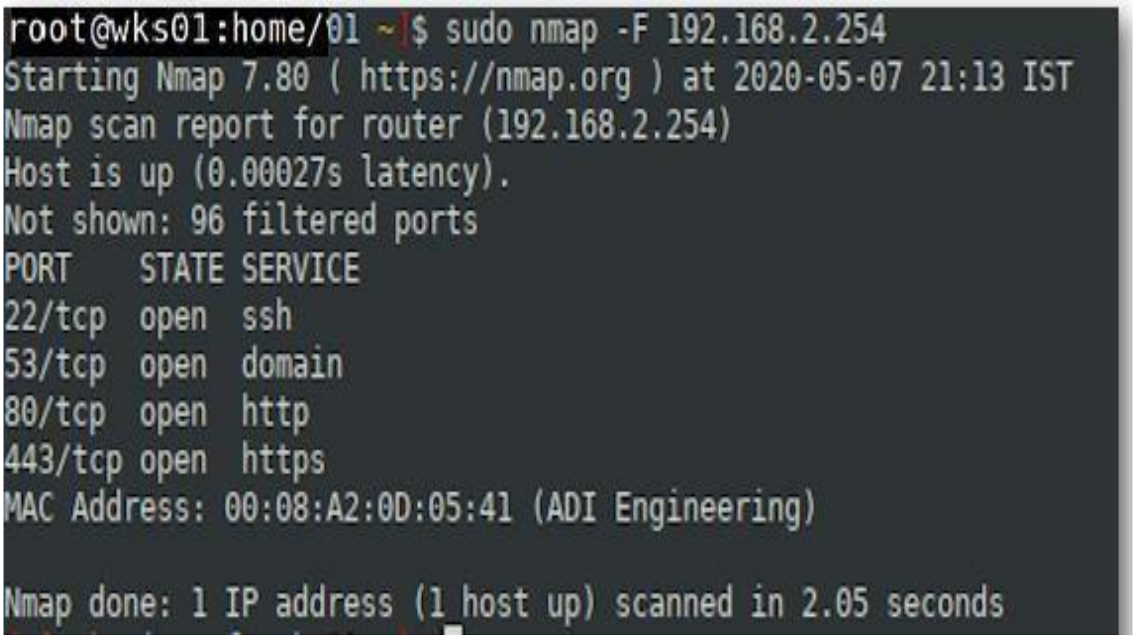
MAC Address: 00:11:32:11:15:FC (Synology Incorporated)

Nmap done: 256 IP addresses (4 hosts up) scanned in 2.80 second

10: How do I perform a fast scan?

nmap -F 192.168.2.254 or

sudo nmap -F 192.168.2.254 [Sample output]



```
root@wks01:home/01 ~]$ sudo nmap -F 192.168.2.254
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-07 21:13 IST
Nmap scan report for router (192.168.2.254)
Host is up (0.00027s latency).
Not shown: 96 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
53/tcp  open  domain
80/tcp  open  http
443/tcp open  https
MAC Address: 00:08:A2:0D:05:41 (ADI Engineering)

Nmap done: 1 IP address (1 host up) scanned in 2.05 seconds
```

11: Display the reason a port is in a particular state

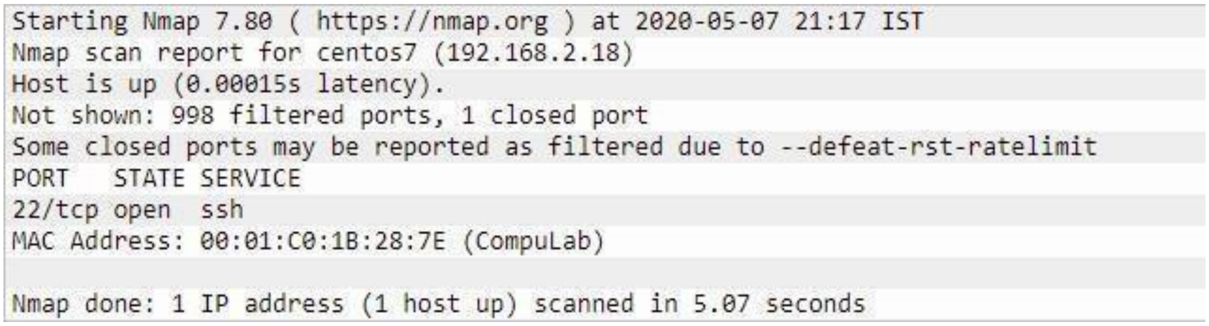
nmap --reason 192.168.1.1

nmap --reason server1.cyberciti.biz

12: Only show open (or possibly open) ports

nmap --open server1.cyberciti.biz

nmap --open 192.168.2.18 [Sample output]



```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-07 21:17 IST
Nmap scan report for centos7 (192.168.2.18)
Host is up (0.00015s latency).
Not shown: 998 filtered ports, 1 closed port
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:01:C0:1B:28:7E (CompuLab)

Nmap done: 1 IP address (1 host up) scanned in 5.07 seconds
```

13: Show all packets sent and received

nmap --packet-trace 192.168.1.1

nmap --packet-trace server1.cyberciti.biz

14: Show host interfaces and routes

This is useful for debugging (ip command or route command or netstat command like output using nmap)

nmap --iflist

15: How do I scan specific ports?

nmap -p [port] hostName

nmap -p 80 192.168.1.1 (Scan port 80)

```
nmap -p T:80 192.168.1.1 ( Scan TCP port 80)

nmap -p U:53 192.168.1.1 (Scan UDP port 53)

nmap -p 80,443 192.168.1.1 (Scan two ports)

nmap -p 80-200 192.168.1.1 (Scan port ranges)

nmap -p U:53,111,137,T:21-25,80,139,8080 192.168.1.1 (Combine all options)

nmap -p U:53,111,137,T:21-25,80,139,8080 server1.cyberciti.biz

nmap -v -sU -sT -p U:53,111,137,T:21-25,80,139,8080 192.168.1.254

nmap -p "*" 192.168.1.1 ( Scan all ports with * wildcard)

nmap --top-ports 5 192.168.1.1 (Scan top ports i.e. scan $number most common ports)nmap --top-ports 10 192.168.1.1
```

16: The fastest way to scan all your devices/computers for open ports ever

```
nmap -T5 192.168.1.0/24
```

17: How do I detect remote operating system?

You can identify a remote host apps and OS using the -O option:

```
nmap -O 192.168.1.1

nmap -O --osscan-guess 192.168.1.1

nmap -v -O --osscan-guess 192.168.1.1
```

18: How do I detect remote services (server / daemon) version numbers?

```
nmap -sV 192.168.1.1
```

19: Scan a host using TCP ACK (PA) and TCP Syn (PS) ping

If firewall is blocking standard ICMP pings, try the following host discovery methods:

```
nmap -PS 192.168.1.1

nmap -PS 80,21,443 192.168.1.1

nmap -PA 192.168.1.1

nmap -PA 80,21,200-512 192.168.1.1
```

20: Scan a host using IP protocol ping

```
nmap -PO 192.168.1.1
```

21: Scan a host using UDP ping

This scan bypasses firewalls and filters that only screen TCP:

```
nmap -PU 192.168.1.1

nmap -PU 2000.2001 192.168.1.1
```

22: Find out the most commonly used TCP ports using TCP SYN Scan

nmap -sS 192.168.1.1 (Stealthy scan **)**

nmap -sT 192.168.1.1(Find out the most commonly used TCP ports using TCP connect scan (warning: no stealth scan) (OS Fingerprinting **)**

nmap -sA 192.168.1.1(Find out the most commonly used TCP ports using TCP ACK scan)

nmap -sW 192.168.1.1(Find out the most commonly used TCP ports using TCP Window scan)

nmap -sM 192.168.1.1 (Find out the most commonly used TCP ports using TCP Maimon scan)

23: Scan a host for UDP services (UDP scan)

Most popular services on the Internet run over the TCP protocol. DNS, SNMP, and DHCP are three of the most common UDP services. Use the following syntax to find out UDP services:

nmap -sU nas03

nmap -sU 192.168.1.1

24: Scan for IP protocol

This type of scan allows you to determine which IP protocols (TCP, ICMP, IGMP, etc.) are supported by target machines:

nmap -sO 192.168.1.1

25: Scan a firewall for security weakness

The following scan types exploit a subtle loophole in the TCP and good for testing security of common attacks:

nmap -sN 192.168.1.254 (TCP Null Scan to fool a firewall to generate a response.Does not set any bits [TCP flag header is 0])

nmap -sF 192.168.1.254 (TCP Fin scan to check firewall. Sets just the TCP FIN bit)

nmap -sX 192.168.1.254 (TCP Xmas scan to check firewall. Sets the FIN, PSH, and URG flags, lighting the packet up like a Christmas tree)

See how to block Xmas packkets, syn-floods and other conman attacks with iptables.

26: Scan a firewall for packets fragments

The -f option causes the requested scan (including ping scans) to use tiny fragmented IP packets. The idea is to split up the TCP header over several packets to make it harder for packet filters, intrusion detection systems, and other annoyances to detect what you are doing.

nmap -f 192.168.1.1

nmap -f fw2.nixcraft.net.in

nmap -f 15 fw2.nixcraft.net.in

nmap --mtu 32 192.168.1.1(Set your own offset size with the --mtu option)

27: Cloak a scan with decoys

The -D option it appear to the remote host that the host(s) you specify as decoys are scanning the target network too. Thus their IDS might report 5-10 port scans from unique IP addresses, but they won’t know which IP was scanning them and which were innocent decoys:

nmap -n -Ddecoy-ip1,decoy-ip2,your-own-ip,decoy-ip3,decoy-ip4 remote-
nmap -n -D192.168.1.5,10.5.1.2,172.1.2.4,3.4.2.1 192.168.1.5

28: Scan a firewall for MAC address spoofing

nmap --spoof-mac MAC-ADDRESS-HERE 192.168.1.1(Spoof your MAC address)

nmap -v -sT -PN --spoof-mac MAC-ADDRESS-HERE 192.168.1.1(Add other options)

nmap -v -sT -PN --spoof-mac 0 192.168.1.1(Use a random MAC address . The number 0, means nmap chooses a completely random MAC address)

29: How do I save output to a text file?

The syntax is:

nmap 192.168.1.1 > output.txt

nmap -oN /path/to/filename 192.168.1.1

nmap -oN output.txt 192.168.1.1

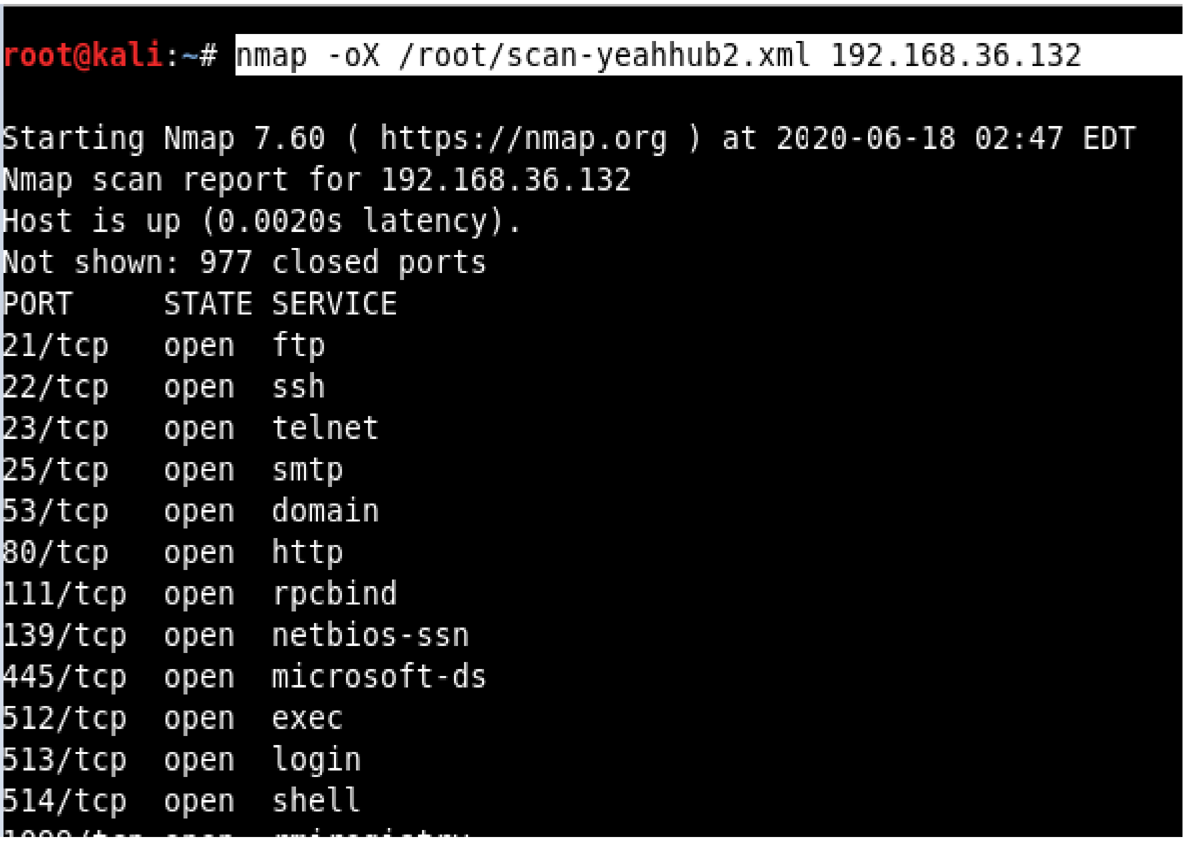
30: Not a fan of command line tools?

Try zenmap the official network mapper front end: Zenmap is the official Nmap Security Scanner GUI. It is a multi-platform (Linux, Windows, Mac OS X, BSD, etc.) free and open source application which aims to make Nmap easy for beginners to use while providing advanced features for experienced Nmap users. Frequently used scans can be saved as profiles to make them easy to run repeatedly. A command creator allows interactive creation of Nmap command lines. Scan results can be saved and viewed later. Saved scan results can be compared with one another to see how they differ. The results of recent scans are stored in a searchable database.

31 How do I save output to a xml file?

Syntax:

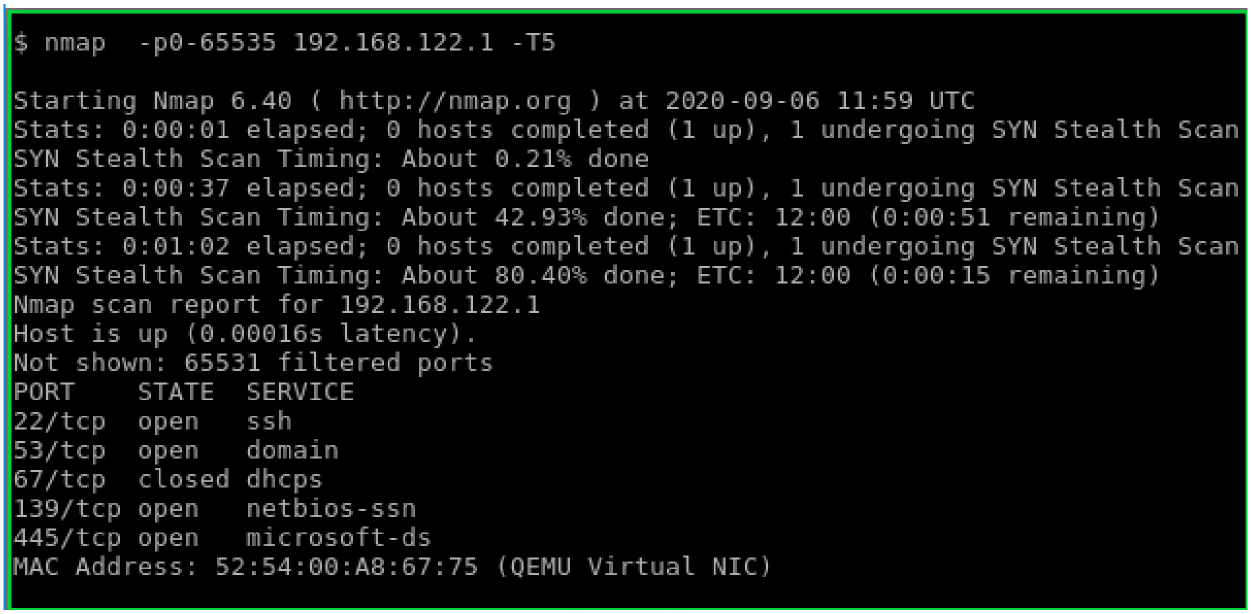
nmap -oX </path/filename.xml> <target>



```
root@kali:~# nmap -oX /root/scan-yeahhub2.xml 192.168.36.132

Starting Nmap 7.60 ( https://nmap.org ) at 2020-06-18 02:47 EDT
Nmap scan report for 192.168.36.132
Host is up (0.0020s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
```

32: Faster Scan For All Ports

$ nmap -p0-65535 192.168.122.1 -T5 [sample output]



```
$ nmap  -p0-65535 192.168.122.1 -T5

Starting Nmap 6.40 ( http://nmap.org ) at 2020-09-06 11:59 UTC
Stats: 0:00:01 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 0.21% done
Stats: 0:00:37 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 42.93% done; ETC: 12:00 (0:00:51 remaining)
Stats: 0:01:02 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 80.40% done; ETC: 12:00 (0:00:15 remaining)
Nmap scan report for 192.168.122.1
Host is up (0.00016s latency).
Not shown: 65531 filtered ports
PORT    STATE  SERVICE
22/tcp  open   ssh
53/tcp  open   domain
67/tcp  closed dhcps
139/tcp open   netbios-ssn
445/tcp open   microsoft-ds
MAC Address: 52:54:00:A8:67:75 (QEMU Virtual NIC)
```

<u>Result:-</u>

Familiarized with NMAP scanning techniques.