

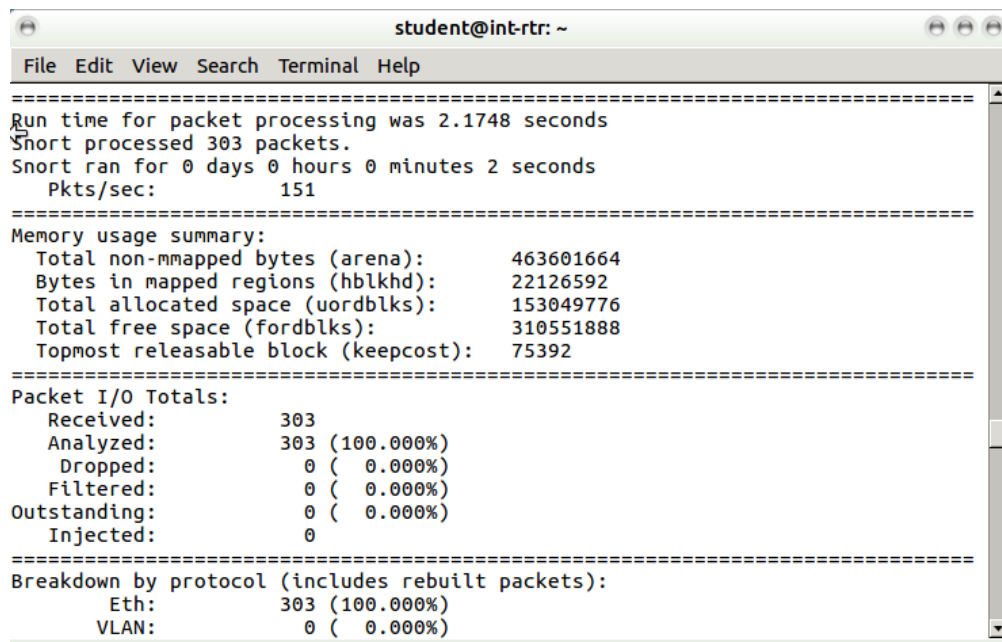
Introduction

Snort can be run in three modes:

1. Sniffer mode: which simply reads the packets off of the network and displays them for you in a continuous stream on the console.
2. Packet logged mode: which logs the packets to disk.
3. Network Intrusion Detection system (NIDS): which performs detection and analysis on network traffic. This is the most complex and configurable mode. In this lab, you will use the NIDS mode.

Lab

1. **List the alerts (from the alerts) and list the corresponding Generator ID, Snort ID and Revision ID of each alert and their significance. (20 points)**
the command for test mode is:
`sudo snort -dev -A test -c /etc/snort/etc/snort.conf -r /home/student/snort_src/ InfectedPcaps/infected.pcap`



```
student@int-rtr: ~  
File Edit View Search Terminal Help  
=====
```

Run time for packet processing was 2.1748 seconds	
Snort processed 303 packets.	
Snort ran for 0 days 0 hours 0 minutes 2 seconds	
Pkts/sec:	151

```
=====
```

Memory usage summary:

Total non-mapped bytes (arena):	463601664
Bytes in mapped regions (hblkhd):	22126592
Total allocated space (uordblks):	153049776
Total free space (fordblks):	310551888
Topmost releasable block (keepcost):	75392

```
=====
```

Packet I/O Totals:

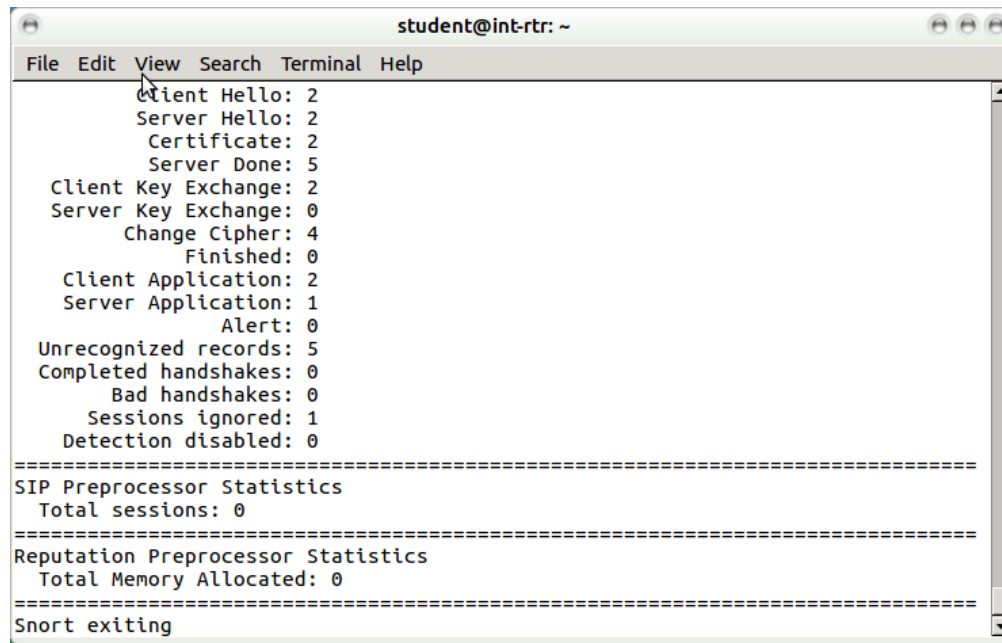
Received:	303
Analyzed:	303 (100.000%)
Dropped:	0 (0.000%)
Filtered:	0 (0.000%)
Outstanding:	0 (0.000%)
Injected:	0

```
=====
```

Breakdown by protocol (includes rebuilt packets):

Eth:	303 (100.000%)
VLAN:	0 (0.000%)

Figure 1: test mode snort and pcap



The screenshot shows a terminal window titled 'student@int-rtr: ~'. The window contains the output of Snort in test mode. The output lists various protocol events and their counts, followed by statistics for SIP and Reputation preprocessors, and ends with 'Snort exiting'.

```
student@int-rtr: ~
File Edit View Search Terminal Help
Client Hello: 2
Server Hello: 2
Certificate: 2
Server Done: 5
Client Key Exchange: 2
Server Key Exchange: 0
Change Cipher: 4
Finished: 0
Client Application: 2
Server Application: 1
Alert: 0
Unrecognized records: 5
Completed handshakes: 0
Bad handshakes: 0
Sessions ignored: 1
Detection disabled: 0
=====
SIP Preprocessor Statistics
Total sessions: 0
=====
Reputation Preprocessor Statistics
Total Memory Allocated: 0
=====
Snort exiting
```

Figure 2: test mode snort and pcap

After test mode, we can vim the alert file, and figure out the three alerts:

- (a) the first two
[1 : 25042 : 4]
Generator ID: 1 → snort general alert
Snort ID: 16669 → will be generated when an attempt is made to exploit a known vulnerability in jdk.
Revision ID: 4 → This denotes the number of times an alert is revised.
- (b) the last one
[1 : 16669 : 5]
Generator ID: 1 → snort general alert
Snort ID: 16669 → will be generated when a spyware application related activity is detected.
Revision ID: 5 → This denotes the number of times an alert is revised.

Also, if we use the full mode for the snort, we will get the Figure 4, with more details.

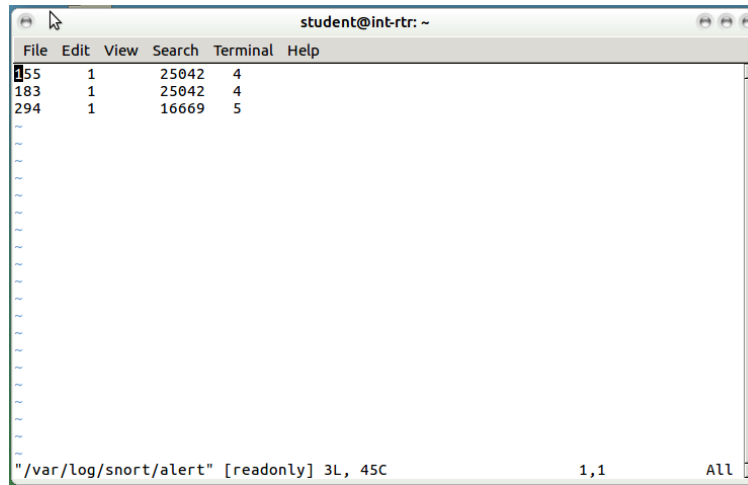


Figure 3: alert after test mode

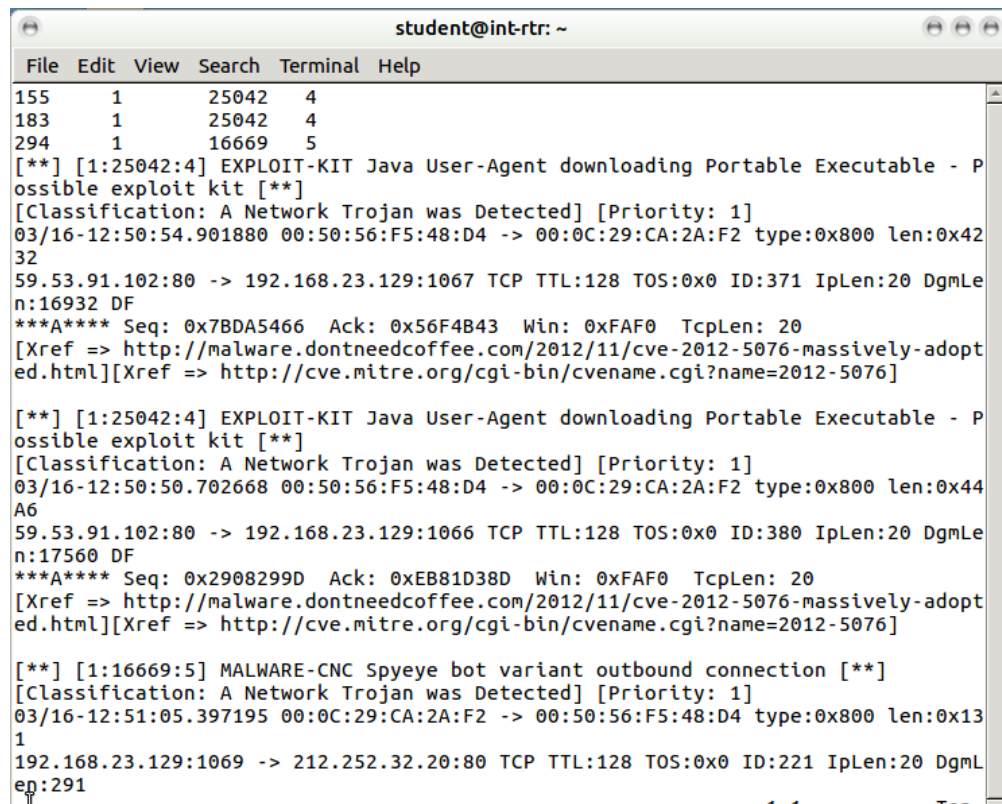


Figure 4: alert after full mode

2. The alert file contains the output when the file was run using the “-A test” option. This will display the packet numbers of the corresponding packets that triggered alerts. Use Wireshark and locate these packets. List the source and destination IP address, source and destination port numbers and protocol used for each packet. (15 points)

- (a) packet 155
source IP address: 192.168.23.129
destination IP address: 59.53.91.102
source port numbers: 1067
destination port numbers: 80
protocol: TCP



Figure 5: packet 155

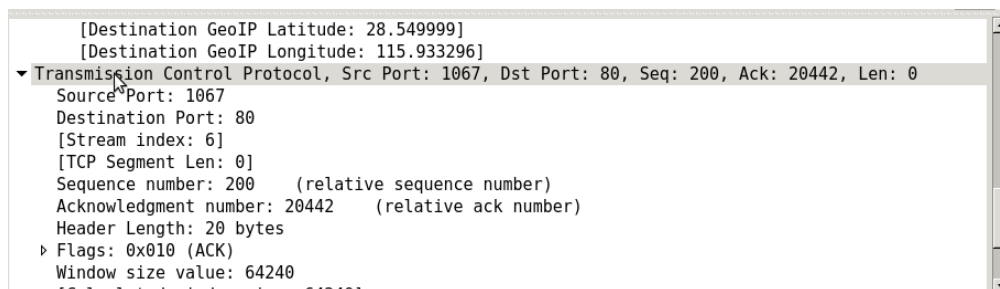


Figure 6: packet 155

- (b) packet 183
source IP address: 192.168.23.129
destination IP address: 59.53.91.102
source port numbers: 1067

destination port numbers: 80
protocol: TCP

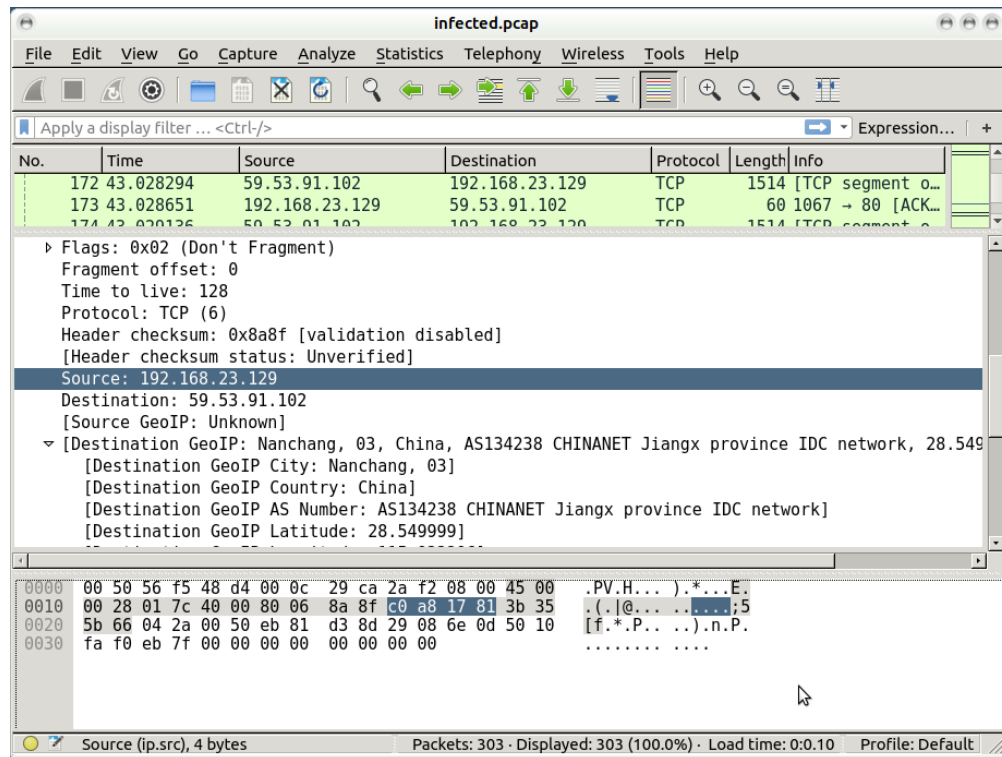


Figure 7: packet 183

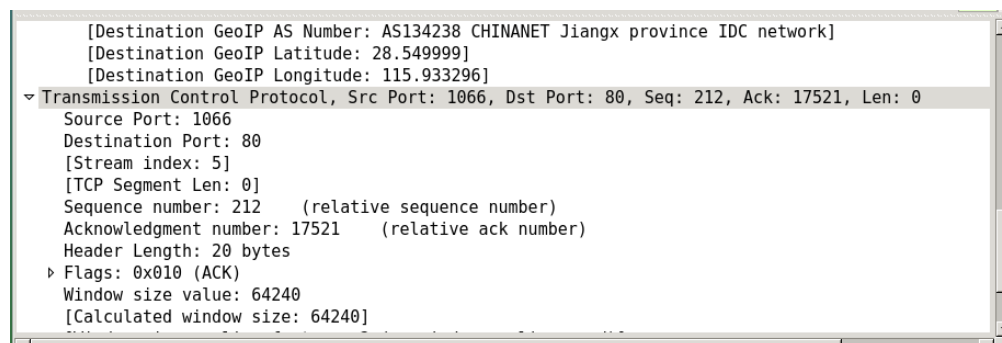


Figure 8: packet 183

- (c) packet 294
source IP address: 212.252.32.20
destination IP address: 192.168.23.129
source port numbers: 80
destination port numbers: 1069
protocol: TCP

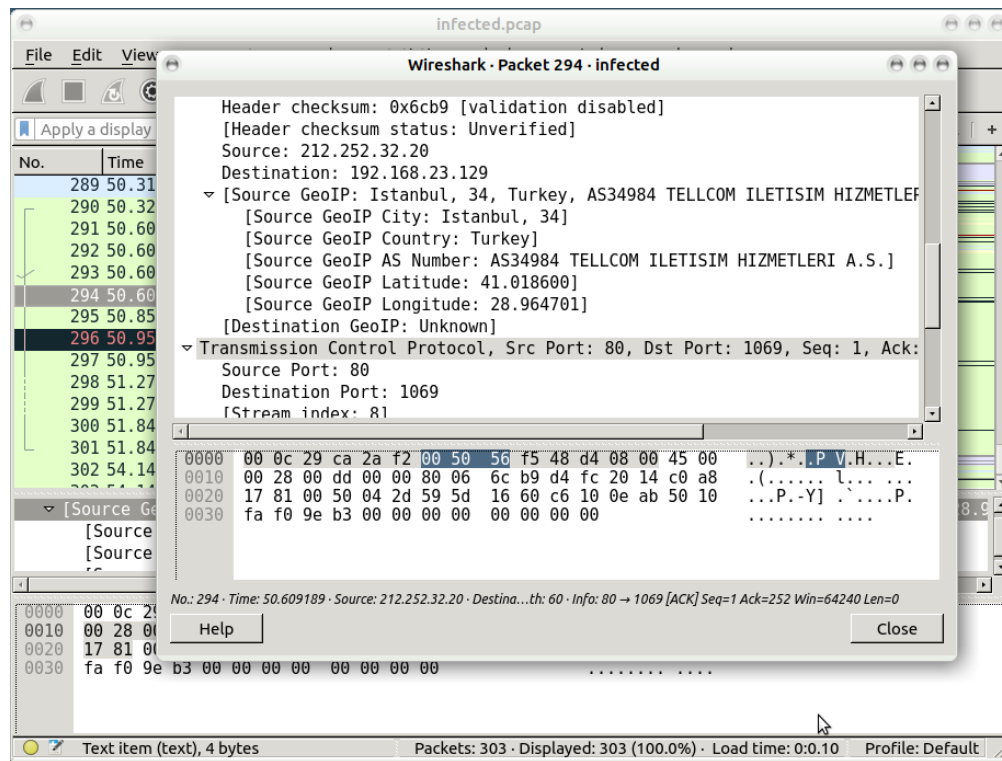


Figure 9: packet 294

3. Use a filter and list all the DNS queries and the resolved IP addresses. Include the filter in the lab write up. (15 points)
We can directly use 'dns' to filter.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.23.129	192.168.23.2	DNS	68	Standard query 0xfalc A nrtj...
2	0.988900	192.168.23.129	192.168.23.2	DNS	68	Standard query 0xfalc A nrtj...
3	1.987301	192.168.23.129	192.168.23.2	DNS	68	Standard query 0xfalc A nrtj...
4	2.909144	192.168.23.2	192.168.23.129	DNS	162	Standard query response 0xfa...
6	2.929185	192.168.23.2	192.168.23.129	DNS	162	Standard query response 0xfa...
7	2.930238	192.168.23.2	192.168.23.129	DNS	162	Standard query response 0xfa...
43	19.900252	192.168.23.129	192.168.23.2	DNS	68	Standard query 0x5b1d A nrtj...
44	19.971014	192.168.23.2	192.168.23.129	DNS	162	Standard query response 0x5b...
90	29.821145	192.168.23.129	192.168.23.2	DNS	85	Standard query 0xe78a PTR 10...
93	30.666108	192.168.23.2	192.168.23.129	DNS	143	Standard query response 0xe7...
288	50.210596	192.168.23.129	192.168.23.2	DNS	71	Standard query 0xbca3 A free...
289	50.310134	192.168.23.2	192.168.23.129	DNS	235	Standard query response 0xbca...

[Header checksum status: Unverified]
Source: 192.168.23.2
Destination: 192.168.23.129
[Source GeoIP: Unknown]
[Destination GeoIP: Unknown]

0010 00 dd 00 db 00 00 80 11 89 61 c0 a8 17 02 c0 a8a.....
0020 17 81 00 35 cd 13 00 c9 c3 2d bc a3 81 80 00 01 ...5.....
0030 00 01 00 04 00 04 08 66 72 65 65 77 61 79 73 02f reeways.
0040 69 6e 00 00 01 00 01 c0 0c 00 01 00 01 00 00 00 in.....
0050 05 00 04 d4 fc 20 14 c0 0c 00 02 00 01 00 00 00
0060 05 00 12 03 6e 73 33 08 65 76 65 72 79 64 6e 73ns3. everydns
0070 03 6e 65 74 00 c0 0c 00 02 00 01 00 00 00 05 00 .net.....

Domain Name System: Protocol Packets: 303 · Displayed: 12 (4.0%) · Load time: 0:0.11 Profile: Default

Figure 10: dns queries

4. There were HTTP sessions established to download 2 java applets. What were the names of the two .jar files that implemented these applets? (10 points)

We can use 'http' to filter and then we will notice that there are two jar files: q.jar and sdfg.jar.

No.	Time	Source	Destination	Protocol	Length	Info
10	3.576662	192.168.23.129	59.53.91.102	HTTP	517	GET /true.php HTTP/1.1
13	6.480119	59.53.91.102	192.168.23.129	HTTP	136	HTTP/1.1 200 OK (text...
15	6.518319	192.168.23.129	59.53.91.102	HTTP	364	GET /xxx.xxx HTTP/1.1
32	7.209846	59.53.91.102	192.168.23.129	HTTP	478	HTTP/1.1 200 OK (text...
49	20.485308	192.168.23.129	59.53.91.102	HTTP	309	GET /favicon.ico HTTP/...
55	23.557198	59.53.91.102	192.168.23.129	HTTP	631	HTTP/1.1 404 Not Found...
62	23.685217	192.168.23.129	59.53.91.102	HTTP	314	GET /q.jar HTTP/1.1
64	23.712064	192.168.23.129	59.53.91.102	HTTP	317	GET /sdfg.jar HTTP/1.1
85	29.268989	59.53.91.102	192.168.23.129	HTTP	90	HTTP/1.1 200 OK (appl...
98	34.066512	59.53.91.102	192.168.23.129	HTTP	1504	HTTP/1.1 200 OK (appl...
105	34.894795	192.168.23.129	59.53.91.102	HTTP	265	GET //loading.php?spl=...
115	38.794966	192.168.23.129	59.53.91.102	HTTP	253	GET //loading.php?spl=...
217	43.893260	59.53.91.102	192.168.23.129	HTTP	228	HTTP/1.1 200 OK (appl...
273	46.484170	59.53.91.102	192.168.23.129	HTTP	432	HTTP/1.1 200 OK (appl...
293	50.609172	192.168.23.129	212.252.32.20	HTTP	305	GET /11111/gate.php?gu...

Source: 192.168.23.129
Destination: 59.53.91.102
[Source GeoIP: Unknown]
[Destination GeoIP: Nanhang, 02, China, AS134228, CHINANET Jiangsu province IDC network, 28, 540000, 115]

0010 01 2c 01 4d 40 00 80 06 89 ba c0 a8 17 81 3b 35 ...Me...;5
0020 5b 66 04 29 00 50 91 4c 33 e0 4b 16 2e 3b 50 18 [(f.)P.L 3.K.;P.
0030 fa f0 32 35 00 00 47 45 54 20 2f 71 2e 6a 61 72 ..25..GE T /q.jar
0040 20 48 54 54 50 2f 31 2e 31 0d 0a 61 63 63 65 70 HTTP/1. 1..accep
0050 74 2d 65 6e 63 6f 64 69 6e 67 3a 20 70 61 63 6b t-encodi ng: pack
0060 32 30 30 2d 67 7a 69 70 2c 20 67 7a 69 70 0d 0a 200-gzip , gzip..
0070 63 6f 6e 74 65 6e 74 2d 74 79 70 65 3a 20 61 70 content- type: ap

Text item (text), 4 bytes Packets: 303 · Displayed: 16 (5.3%) · Load time: 0:0.14 Profile: Default

Figure 11: two download jar file

5. As part of the infection, a malicious executable file was downloaded onto the client's computer. What was the file's MD5 hash? Hint: It ends on "91ed". (10 points)

According to the hint, the md5 will end with '91ed', and packet 293 has that. And the md5 hash is: 5942ba36cf732097479c51986eee91ed

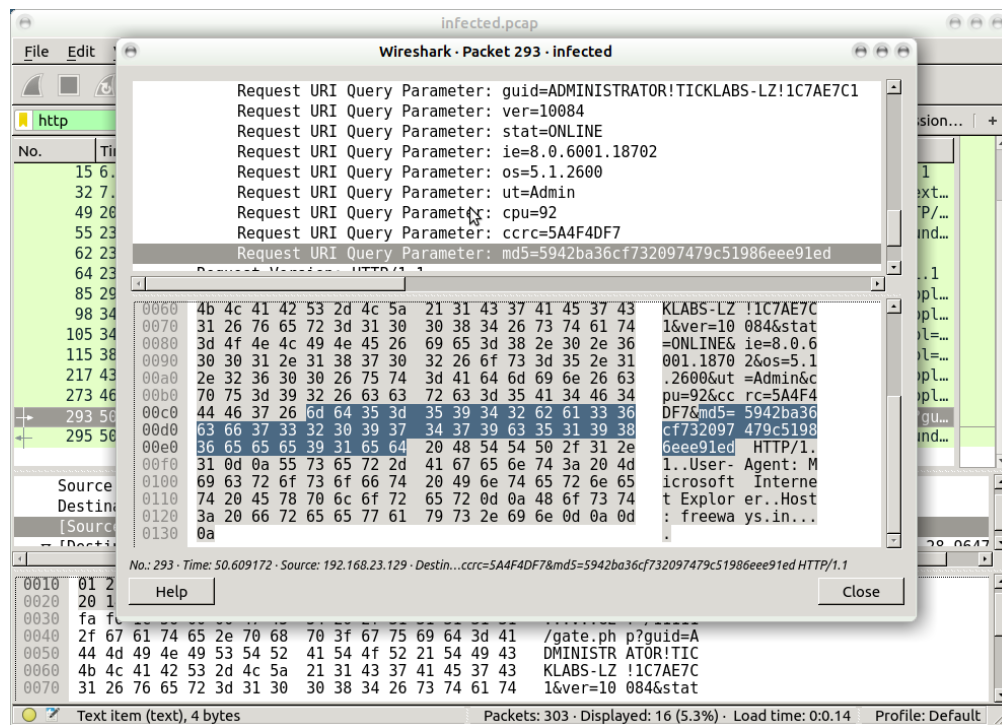


Figure 12: md5 hash

6. Which browser is being used by the client? (5 points)

The client uses Microsoft Internet Explorer.