Table of content

Practical - 1

- **Aim:**
  To analyze TCP connections using Wireshark and understand how data packets are transmitted over a network.

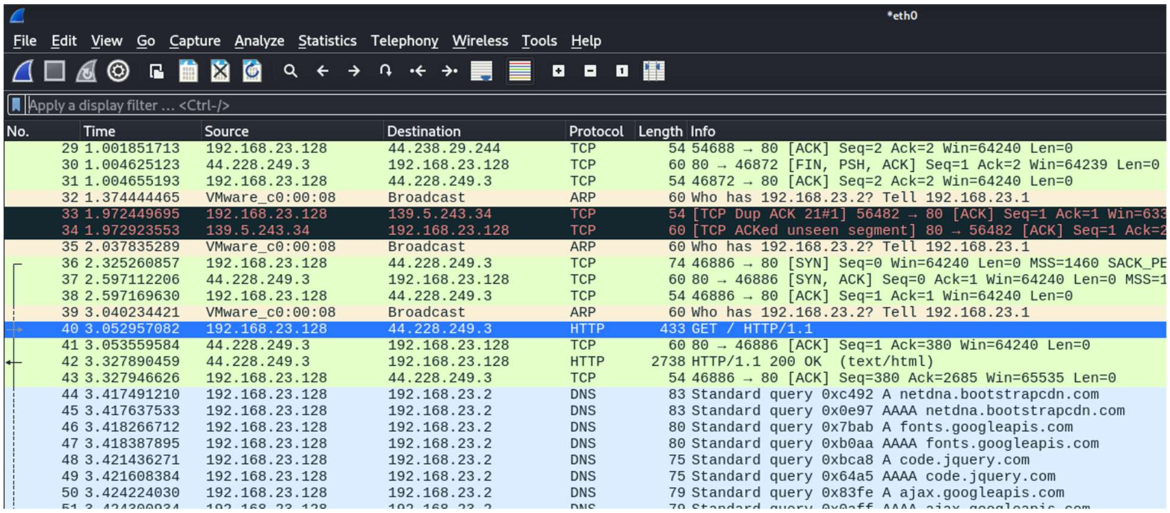
- **Tool/Application Used:**
  Wireshark

- **Theory:**
  TCP (Transmission Control Protocol) is a connection-oriented protocol that ensures reliable data transfer between devices. It establishes connections using a three-way handshake process. Wireshark is a powerful network protocol analyzer that captures and displays data packets, enabling the analysis of network traffic. By capturing packets, we can identify the TCP handshake, sequence numbers, acknowledgments, and various TCP flags (SYN, ACK, FIN, etc.).

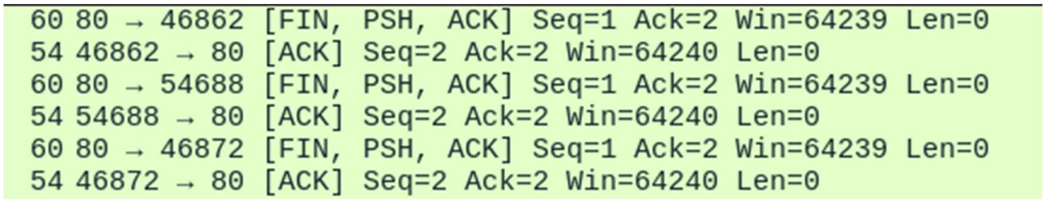
- **Procedure:**

  1. Open Wireshark and start a network capture on the desired network interface.
  2. Perform an action that generates TCP traffic, such as opening a website or connecting to a server.
  3. Stop the capture after sufficient data is collected.
  4. Use the filter tcp in Wireshark to display only TCP packets.
  5. Locate the three-way handshake process by identifying packets with SYN, SYN-ACK, and ACK flags.
  6. Observe the sequence and acknowledgment numbers to understand the flow of communication.
  7. Analyze any retransmissions, delays, or packet loss in the TCP session.
  8. Save the capture file for reporting purposes.

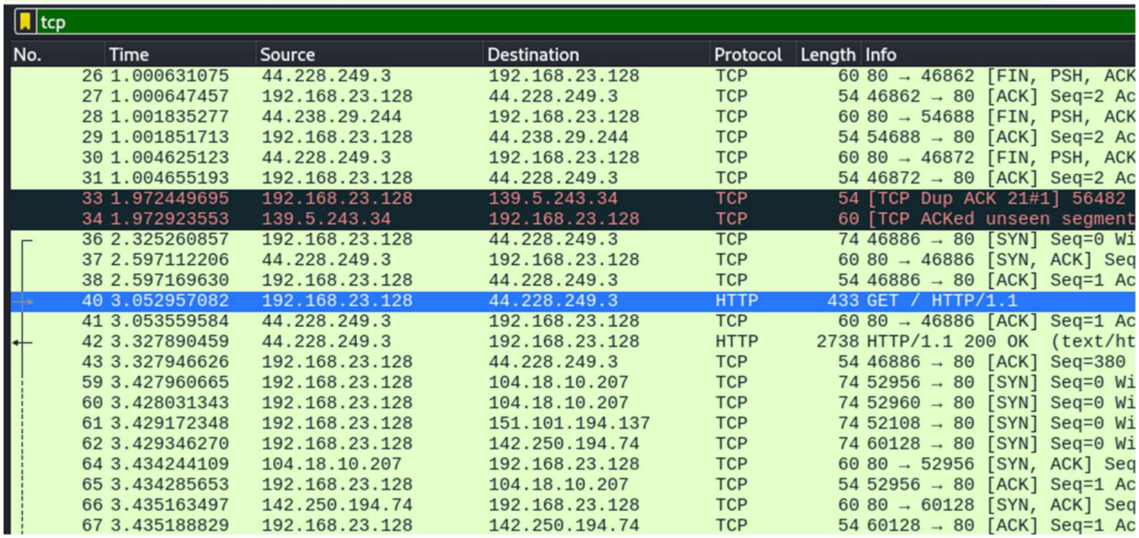
- **Output:**



```
No.   Time         Source           Destination      Protocol Length Info
   29 1.001851713  192.168.23.128   44.238.29.244    TCP        54 54688 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
   30 1.004625123  44.228.249.3     192.168.23.128   TCP        60 80 → 46872 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
   31 1.004655193  192.168.23.128   44.228.249.3     TCP        54 46872 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
   32 1.374444465  VMware_c0:00:08  Broadcast        ARP        60 Who has 192.168.23.2? Tell 192.168.23.1
   33 1.972449695  192.168.23.128   139.5.243.34     TCP        54 [TCP Dup ACK 21#1] 56482 → 80 [ACK] Seq=1 Ack=1 Win=633
   34 1.972923553  139.5.243.34     192.168.23.128   TCP        60 [TCP ACKed unseen segment] 80 → 56482 [ACK] Seq=1 Ack=2
   35 2.037835289  VMware_c0:00:08  Broadcast        ARP        60 Who has 192.168.23.2? Tell 192.168.23.1
   36 2.325260857  192.168.23.128   44.228.249.3     TCP        74 46886 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PE
   37 2.597112206  44.228.249.3     192.168.23.128   TCP        60 80 → 46886 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1
   38 2.597169630  192.168.23.128   44.228.249.3     TCP        54 46886 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
   39 3.040234421  VMware_c0:00:08  Broadcast        ARP        60 Who has 192.168.23.2? Tell 192.168.23.1
   40 3.052957082  192.168.23.128   44.228.249.3     HTTP      433 GET / HTTP/1.1
   41 3.053559584  44.228.249.3     192.168.23.128   TCP        60 80 → 46886 [ACK] Seq=1 Ack=380 Win=64240 Len=0
   42 3.327890459  44.228.249.3     192.168.23.128   HTTP     2738 HTTP/1.1 200 OK  (text/html)
   43 3.327946626  192.168.23.128   44.228.249.3     TCP        54 46886 → 80 [ACK] Seq=380 Ack=2685 Win=65535 Len=0
   44 3.417491210  192.168.23.128   192.168.23.2     DNS        83 Standard query 0xc492 A netdna.bootstrapcdn.com
   45 3.417637533  192.168.23.128   192.168.23.2     DNS        83 Standard query 0x0e97 AAAA netdna.bootstrapcdn.com
   46 3.418266712  192.168.23.128   192.168.23.2     DNS        80 Standard query 0x7bab A fonts.googleapis.com
   47 3.418387895  192.168.23.128   192.168.23.2     DNS        80 Standard query 0xb0aa AAAA fonts.googleapis.com
   48 3.421436271  192.168.23.128   192.168.23.2     DNS        75 Standard query 0xbca8 A code.jquery.com
   49 3.421608384  192.168.23.128   192.168.23.2     DNS        75 Standard query 0x64a5 AAAA code.jquery.com
   50 3.424224030  192.168.23.128   192.168.23.2     DNS        79 Standard query 0x83fe A ajax.googleapis.com
   51 3.424300934  192.168.23.128   192.168.23.2     DNS        79 Standard query 0x0aff AAAA ajax.googleapis.com
```

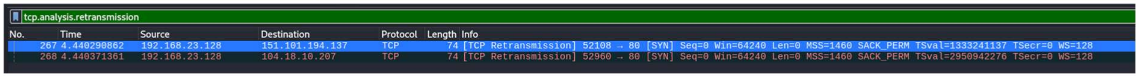


```
60 80 → 46862 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
54 46862 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
60 80 → 54688 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
54 54688 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
60 80 → 46872 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
54 46872 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
```



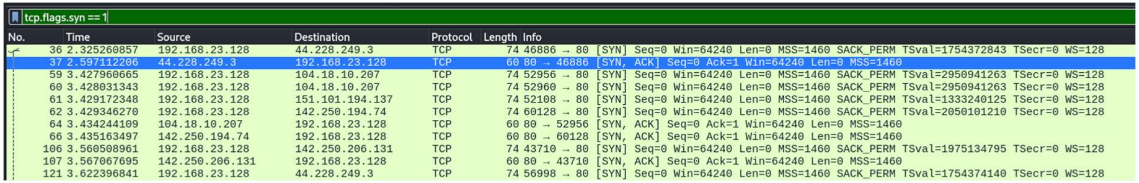
tcp

```
No.    Time         Source           Destination      Protocol Length Info
    26 1.000631075  44.228.249.3     192.168.23.128   TCP        60 80 → 46862 [FIN, PSH, ACK
    27 1.000647457  192.168.23.128   44.228.249.3     TCP        54 46862 → 80 [ACK] Seq=2 Ac
    28 1.001835277  44.238.29.244    192.168.23.128   TCP        60 80 → 54688 [FIN, PSH, ACK
    29 1.001851713  192.168.23.128   44.238.29.244    TCP        54 54688 → 80 [ACK] Seq=2 Ac
    30 1.004625123  44.228.249.3     192.168.23.128   TCP        60 80 → 46872 [FIN, PSH, ACK
    31 1.004655193  192.168.23.128   44.228.249.3     TCP        54 46872 → 80 [ACK] Seq=2 Ac
    33 1.972449695  192.168.23.128   139.5.243.34     TCP        54 [TCP Dup ACK 21#1] 56482
    34 1.972923553  139.5.243.34     192.168.23.128   TCP        60 [TCP ACKed unseen segment
    36 2.325260857  192.168.23.128   44.228.249.3     TCP        74 46886 → 80 [SYN] Seq=0 Wi
    37 2.597112206  44.228.249.3     192.168.23.128   TCP        60 80 → 46886 [SYN, ACK] Seq
    38 2.597169630  192.168.23.128   44.228.249.3     TCP        54 46886 → 80 [ACK] Seq=1 Ac
    40 3.052957082  192.168.23.128   44.228.249.3     HTTP      433 GET / HTTP/1.1
    41 3.053559584  44.228.249.3     192.168.23.128   TCP        60 80 → 46886 [ACK] Seq=1 Ac
    42 3.327890459  44.228.249.3     192.168.23.128   HTTP     2738 HTTP/1.1 200 OK  (text/ht
    43 3.327946626  192.168.23.128   44.228.249.3     TCP        54 46886 → 80 [ACK] Seq=380
    59 3.427960665  192.168.23.128   104.18.10.207    TCP        74 52956 → 80 [SYN] Seq=0 Wi
    60 3.428031343  192.168.23.128   104.18.10.207    TCP        74 52960 → 80 [SYN] Seq=0 Wi
    61 3.429172348  192.168.23.128   151.101.194.137  TCP        74 52108 → 80 [SYN] Seq=0 Wi
    62 3.429346270  192.168.23.128   142.250.194.74   TCP        74 60128 → 80 [SYN] Seq=0 Wi
    64 3.434244109  104.18.10.207    192.168.23.128   TCP        60 80 → 52956 [SYN, ACK] Seq
    65 3.434285653  192.168.23.128   104.18.10.207    TCP        54 52956 → 80 [ACK] Seq=1 Ac
    66 3.435163497  142.250.194.74   192.168.23.128   TCP        60 80 → 60128 [SYN, ACK] Seq
    67 3.435188829  192.168.23.128   142.250.194.74   TCP        54 60128 → 80 [ACK] Seq=1 Ac
```



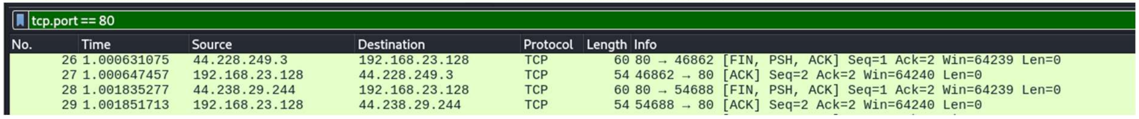
tcp.analysis.retransmission

```
No.   Time         Source          Destination      Protocol Length Info
  267 4.440290862  192.168.23.128  151.101.194.137  TCP        74 [TCP Retransmission] 52108 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1333241137 TSecr=0 WS=128
  268 4.440371361  192.168.23.128  104.18.10.207    TCP        74 [TCP Retransmission] 52960 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2950942276 TSecr=0 WS=128
```



tcp.flags.syn == 1

```
No.   Time         Source           Destination      Protocol Length Info
   36 2.325260857  192.168.23.128   44.228.249.3     TCP        74 46886 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1754372843 TSecr=0 WS=128
   37 2.597112206  44.228.249.3     192.168.23.128   TCP        60 80 → 46886 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
   59 3.427960665  192.168.23.128   104.18.10.207    TCP        74 52956 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2950941263 TSecr=0 WS=128
   60 3.428031343  192.168.23.128   104.18.10.207    TCP        74 52960 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2950941263 TSecr=0 WS=128
   61 3.429172348  192.168.23.128   151.101.194.137  TCP        74 52108 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1333240125 TSecr=0 WS=128
   62 3.429346270  192.168.23.128   142.250.194.74   TCP        74 60128 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2050101210 TSecr=0 WS=128
   64 3.434244109  104.18.10.207    192.168.23.128   TCP        60 80 → 52956 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
   66 3.435163497  142.250.194.74   192.168.23.128   TCP        60 80 → 60128 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
  106 3.560508961  192.168.23.128   142.250.206.131  TCP        74 43710 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1975134795 TSecr=0 WS=128
  107 3.567067695  142.250.206.131  192.168.23.128   TCP        60 80 → 43710 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
  121 3.622396841  192.168.23.128   44.228.249.3     TCP        74 56998 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1754374140 TSecr=0 WS=128
```



tcp.port == 80

```
No.   Time         Source          Destination     Protocol Length Info
   26 1.000631075  44.228.249.3    192.168.23.128  TCP        60 80 → 46862 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
   27 1.000647457  192.168.23.128  44.228.249.3    TCP        54 46862 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
   28 1.001835277  44.238.29.244   192.168.23.128  TCP        60 80 → 54688 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
   29 1.001851713  192.168.23.128  44.238.29.244   TCP        54 54688 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
```

Practical - 2

- **Aim:**
  To capture and analyze HTTP traffic in Wireshark, focusing on GET and POST requests.

- **Tool/Application Used:**
  Wireshark

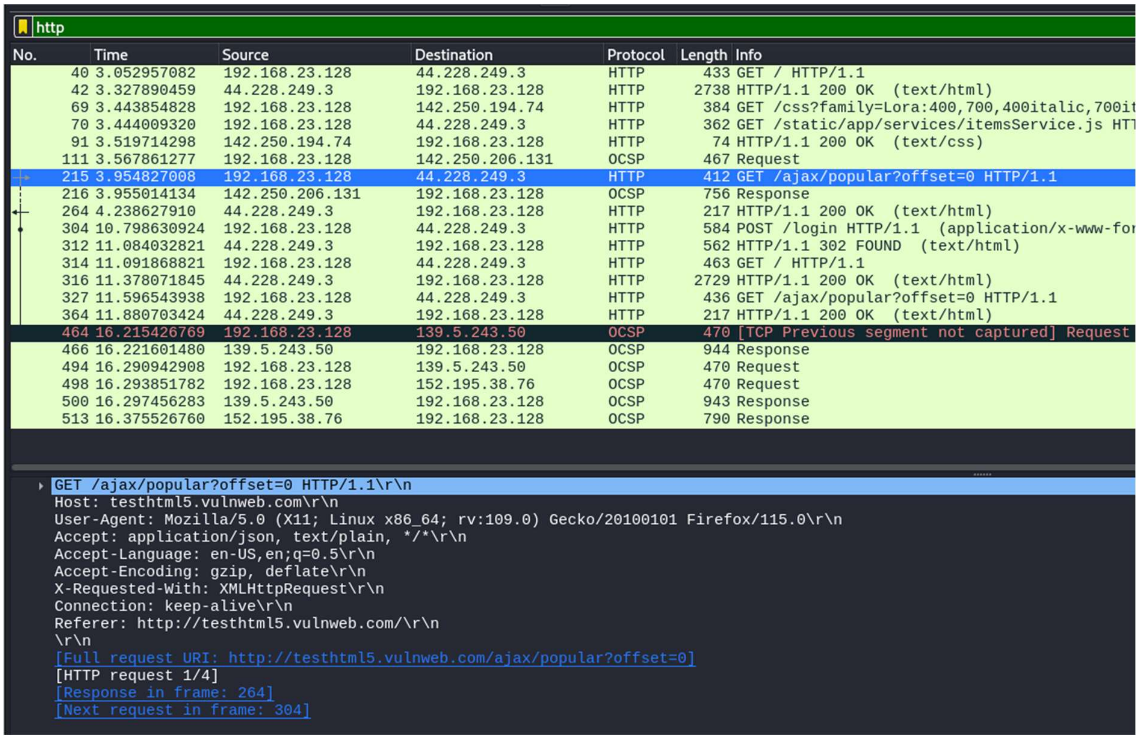
- **Theory:**
  HTTP (Hypertext Transfer Protocol) is a client-server communication protocol used for transmitting data over the web. A GET request is used to retrieve data from a server, while a POST request is used to send data to a server, often for submission forms or APIs. Wireshark allows for capturing and inspecting HTTP traffic, providing detailed insights into request headers, response codes, and payloads.

- **Procedure:**

  1. Open Wireshark and start a network capture on the relevant network interface.
  2. Perform web activities that generate GET and POST requests:
     - Open a browser and access a website (generates a GET request).
     - Fill out and submit a form on a website (generates a POST request).
  3. Stop the capture once sufficient traffic is collected.
  4. Apply the filter http in Wireshark to display only HTTP packets.
  5. Locate GET and POST requests by inspecting the **Info** column for keywords like "GET /" and "POST /".
  6. Select a GET request packet:
     - Examine the **Hypertext Transfer Protocol** section to view request headers, requested resource, and server response.
  7. Select a POST request packet:
     - Analyze the **Hypertext Transfer Protocol** section for headers and any included payload data.
  8. Record the details such as URLs, request methods, and response codes for documentation.

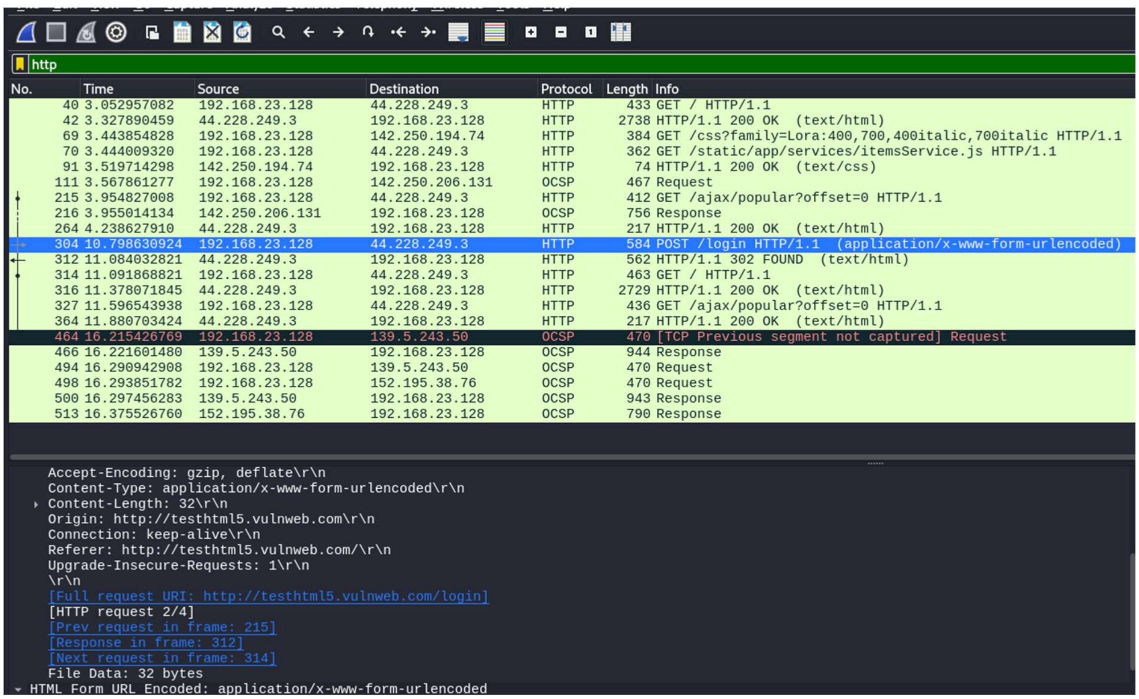
- **Output:**

GET



```
http
No.   Time         Source           Destination      Protocol Length Info
   40 3.052957082  192.168.23.128   44.228.249.3     HTTP      433 GET / HTTP/1.1
   42 3.327890459  44.228.249.3     192.168.23.128   HTTP     2738 HTTP/1.1 200 OK  (text/html)
   69 3.443854828  192.168.23.128   142.250.194.74   HTTP      384 GET /css?family=Lora:400,700,400italic,700it
   70 3.444009320  192.168.23.128   44.228.249.3     HTTP      362 GET /static/app/services/itemsService.js HTT
   91 3.519714298  142.250.194.74   192.168.23.128   HTTP       74 HTTP/1.1 200 OK  (text/css)
  111 3.567861277  192.168.23.128   142.250.206.131  OCSP      467 Request
  215 3.954827008  192.168.23.128   44.228.249.3     HTTP      412 GET /ajax/popular?offset=0 HTTP/1.1
  216 3.955014134  142.250.206.131  192.168.23.128   OCSP      756 Response
  264 4.238627910  44.228.249.3     192.168.23.128   HTTP      217 HTTP/1.1 200 OK  (text/html)
  304 10.798630924 192.168.23.128   44.228.249.3     HTTP      584 POST /login HTTP/1.1  (application/x-www-for
  312 11.084032821 44.228.249.3     192.168.23.128   HTTP      562 HTTP/1.1 302 FOUND  (text/html)
  314 11.091868821 192.168.23.128   44.228.249.3     HTTP      463 GET / HTTP/1.1
  316 11.378071845 44.228.249.3     192.168.23.128   HTTP     2729 HTTP/1.1 200 OK  (text/html)
  327 11.596543938 192.168.23.128   44.228.249.3     HTTP      436 GET /ajax/popular?offset=0 HTTP/1.1
  364 11.880703424 44.228.249.3     192.168.23.128   HTTP      217 HTTP/1.1 200 OK  (text/html)
  464 16.215426769 192.168.23.128   139.5.243.50     OCSP      470 [TCP Previous segment not captured] Request
  466 16.221601480 139.5.243.50     192.168.23.128   OCSP      944 Response
  494 16.290942908 192.168.23.128   139.5.243.50     OCSP      470 Request
  498 16.293851782 192.168.23.128   152.195.38.76    OCSP      470 Request
  500 16.297456283 139.5.243.50     192.168.23.128   OCSP      943 Response
  513 16.375526760 152.195.38.76    192.168.23.128   OCSP      790 Response

  ▸ GET /ajax/popular?offset=0 HTTP/1.1\r\n
    Host: testhtml5.vulnweb.com\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0\r\n
    Accept: application/json, text/plain, */*\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    X-Requested-With: XMLHttpRequest\r\n
    Connection: keep-alive\r\n
    Referer: http://testhtml5.vulnweb.com/\r\n
    \r\n
    [Full request URI: http://testhtml5.vulnweb.com/ajax/popular?offset=0]
    [HTTP request 1/4]
    [Response in frame: 264]
    [Next request in frame: 304]
```

POST



```
http
No.   Time         Source           Destination      Protocol Length Info
   40 3.052957082  192.168.23.128   44.228.249.3     HTTP      433 GET / HTTP/1.1
   42 3.327890459  44.228.249.3     192.168.23.128   HTTP     2738 HTTP/1.1 200 OK  (text/html)
   69 3.443854828  192.168.23.128   142.250.194.74   HTTP      384 GET /css?family=Lora:400,700,400italic,700italic HTTP/1.1
   70 3.444009320  192.168.23.128   44.228.249.3     HTTP      362 GET /static/app/services/itemsService.js HTTP/1.1
   91 3.519714298  142.250.194.74   192.168.23.128   HTTP       74 HTTP/1.1 200 OK  (text/css)
  111 3.567861277  192.168.23.128   142.250.206.131  OCSP      467 Request
  215 3.954827008  192.168.23.128   44.228.249.3     HTTP      412 GET /ajax/popular?offset=0 HTTP/1.1
  216 3.955014134  142.250.206.131  192.168.23.128   OCSP      756 Response
  264 4.238627910  44.228.249.3     192.168.23.128   HTTP      217 HTTP/1.1 200 OK  (text/html)
  304 10.798630924 192.168.23.128   44.228.249.3     HTTP      584 POST /login HTTP/1.1  (application/x-www-form-urlencoded)
  312 11.084032821 44.228.249.3     192.168.23.128   HTTP      562 HTTP/1.1 302 FOUND  (text/html)
  314 11.091868821 192.168.23.128   44.228.249.3     HTTP      463 GET / HTTP/1.1
  316 11.378071845 44.228.249.3     192.168.23.128   HTTP     2729 HTTP/1.1 200 OK  (text/html)
  327 11.596543938 192.168.23.128   44.228.249.3     HTTP      436 GET /ajax/popular?offset=0 HTTP/1.1
  364 11.880703424 44.228.249.3     192.168.23.128   HTTP      217 HTTP/1.1 200 OK  (text/html)
  464 16.215426769 192.168.23.128   139.5.243.50     OCSP      470 [TCP Previous segment not captured] Request
  466 16.221601480 139.5.243.50     192.168.23.128   OCSP      944 Response
  494 16.290942908 192.168.23.128   139.5.243.50     OCSP      470 Request
  498 16.293851782 192.168.23.128   152.195.38.76    OCSP      470 Request
  500 16.297456283 139.5.243.50     192.168.23.128   OCSP      943 Response
  513 16.375526760 152.195.38.76    192.168.23.128   OCSP      790 Response

    Accept-Encoding: gzip, deflate\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
  ▸ Content-Length: 32\r\n
    Origin: http://testhtml5.vulnweb.com\r\n
    Connection: keep-alive\r\n
    Referer: http://testhtml5.vulnweb.com/\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://testhtml5.vulnweb.com/login]
    [HTTP request 2/4]
    [Prev request in frame: 215]
    [Response in frame: 312]
    [Next request in frame: 314]
    File Data: 32 bytes
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
```

Practical - 3

- **Aim:**
  To use Wireshark to detect plaintext passwords transmitted over an unsecured HTTP connection.

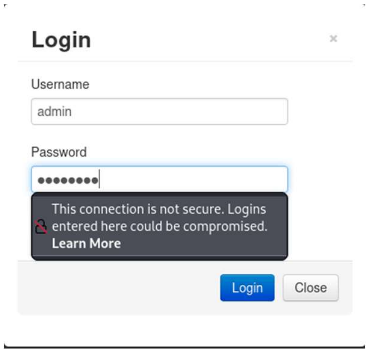
- **Tool/Application Used:**
  Wireshark

- **Theory:**
  HTTP is an unencrypted protocol, meaning data transmitted over it is sent in plaintext. This makes it vulnerable to interception by attackers. Sensitive information such as usernames and passwords transmitted over HTTP can be captured and analyzed using tools like Wireshark. Modern practices recommend using HTTPS to encrypt data in transit.

- **Procedure:**

  1. Open Wireshark and start a network capture on the appropriate network interface.
  2. Access a website using HTTP (not HTTPS). This can be done on a test system or a local web server to ensure ethical practices.
  3. Perform an action that involves logging in, such as entering a username and password in a login form and submitting it.
  4. Stop the capture once the activity is complete.
  5. Apply the filter http to focus only on HTTP packets.
  6. Locate the POST request containing the login information.
     - Look for POST requests in the **Info** column.
  7. Select the POST request packet and inspect the **Hypertext Transfer Protocol** section in the packet details.
     - Look for the **Form Data** or **Parameters** section, where plaintext credentials (username and password) may be visible.
  8. Record the captured credentials for demonstration purposes (on a test setup only).

- **Output:**



Login

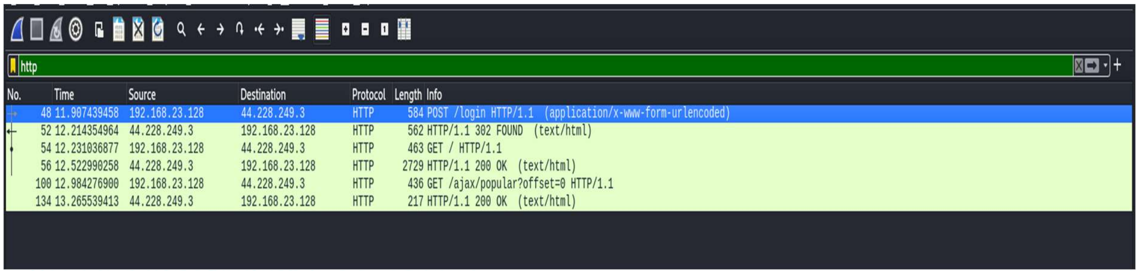
Username

admin

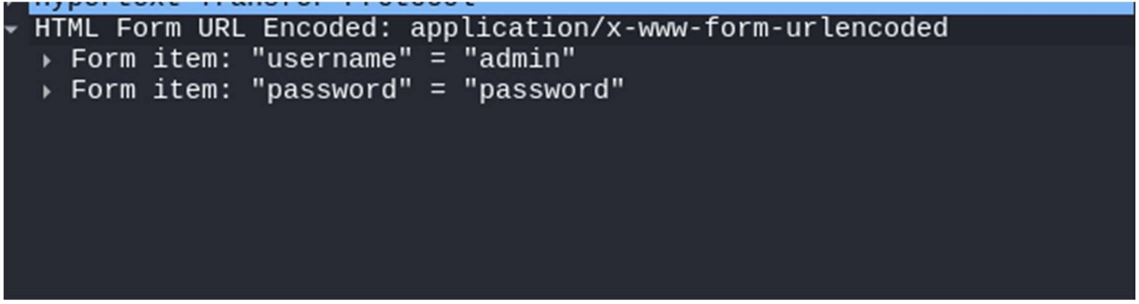
Password

••••••••

This connection is not secure. Logins entered here could be compromised. Learn More



```
http
No.     Time          Source           Destination      Protocol Length Info
     48 11.907439458  192.168.23.128   44.228.249.3     HTTP      584 POST /login HTTP/1.1  (application/x-www-form-urlencoded)
     52 12.214354964  44.228.249.3     192.168.23.128   HTTP      562 HTTP/1.1 302 FOUND  (text/html)
     54 12.231036877  192.168.23.128   44.228.249.3     HTTP      463 GET / HTTP/1.1
     56 12.522990258  44.228.249.3     192.168.23.128   HTTP     2729 HTTP/1.1 200 OK  (text/html)
    100 12.984276900  192.168.23.128   44.228.249.3     HTTP      436 GET /ajax/popular?offset=0 HTTP/1.1
    134 13.265539413  44.228.249.3     192.168.23.128   HTTP      217 HTTP/1.1 200 OK  (text/html)
```



```
Hypertext Transfer Protocol
HTML Form URL Encoded: application/x-www-form-urlencoded
  Form item: "username" = "admin"
  Form item: "password" = "password"
```

Practical - 4

- **Aim:**
  To analyze a .dd case file using Autopsy and generate an investigative report.

- **Tool/Application Used:**
  Autopsy

- **Theory:**
  A .dd file is a raw disk image containing an exact copy of a storage medium's data, including files, directories, and unused space. Forensic tools like Autopsy allow investigators to examine disk images for digital evidence. Autopsy provides features such as timeline analysis, keyword search, file recovery, and metadata extraction to aid forensic investigations.

- **Procedure:**

  1. **Open Autopsy:**
     Launch Autopsy and create a new case. Enter the case name, number, and investigator details.

  2. **Add the Disk Image:**

     - Add the .dd file to the case as a data source.

     - Choose the default ingest modules like file type detection, hash calculation, and keyword search.

  3. **Examine the File System:**

     - Navigate through the disk image to explore files and directories.

     - Look for deleted or hidden files that might contain relevant evidence.

  4. **Search for Artifacts:**

     - Use the **Keyword Search** module to find specific terms or phrases.

     - Analyze artifacts like browser history, emails, and chat logs.

  5. **Generate Timeline:**

     - Use the timeline feature to identify significant events based on file creation, modification, and access times.

  6. **Extract Metadata:**

     - Extract metadata from files to identify their origin, timestamps, and other properties.

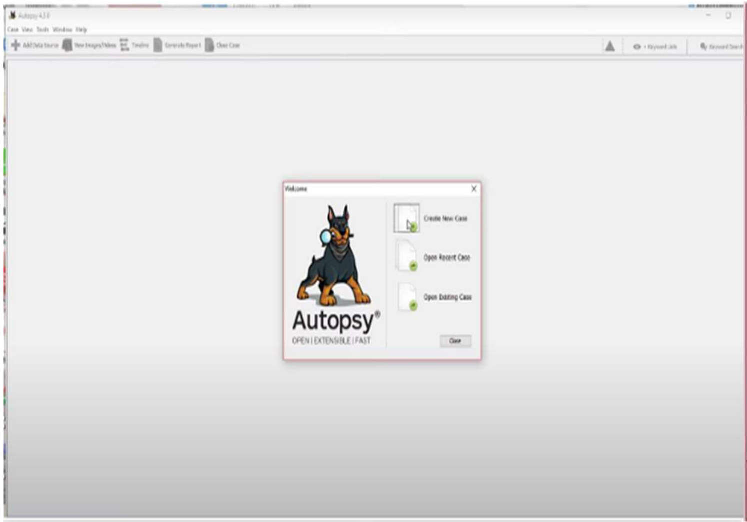
  7. **Document Findings:**

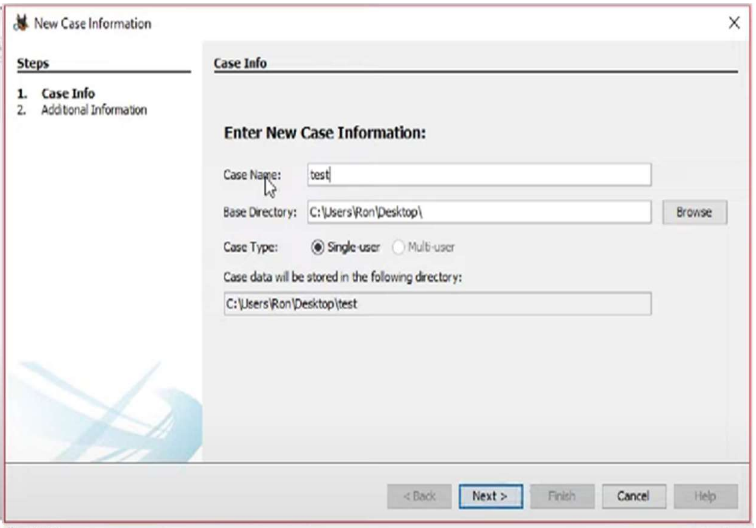
     - Note any suspicious files, keywords, or activities relevant to the investigation.

  8. **Generate a Report:**

     - Use Autopsy’s built-in report generation feature to create an HTML or PDF report summarizing the findings.

Output:




New Case Information

Steps

1. Case Info
2. Additional Information

Case Info

Enter New Case Information:

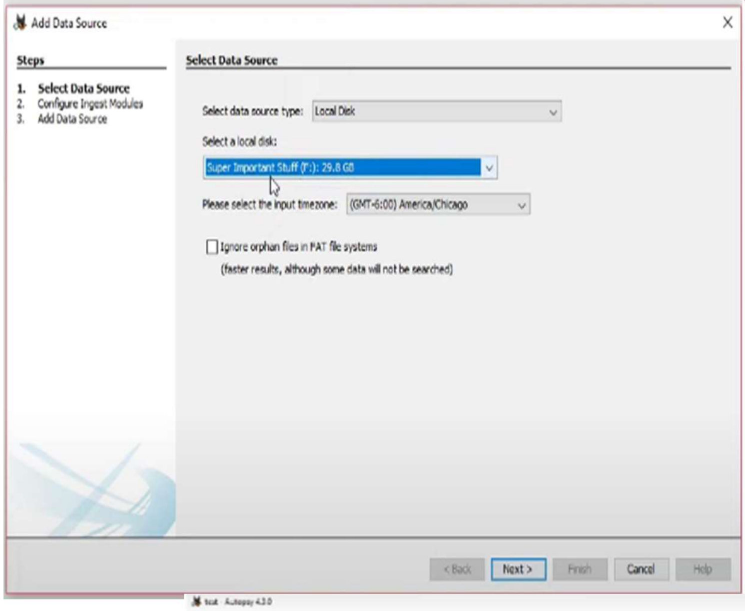
Case Name: test

Base Directory: C:\Users\Ron\Desktop\

Case Type: ● Single-user ○ Multi-user

Case data will be stored in the following directory:

C:\Users\Ron\Desktop\test


Add Data Source

Steps

1. Select Data Source
2. Configure Ingest Modules
3. Add Data Source

Select Data Source

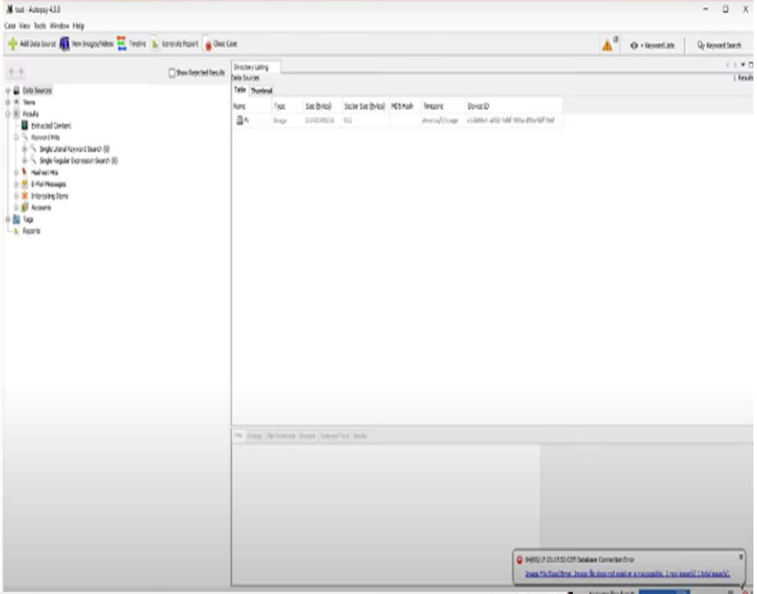
Select data source type: Local Disk

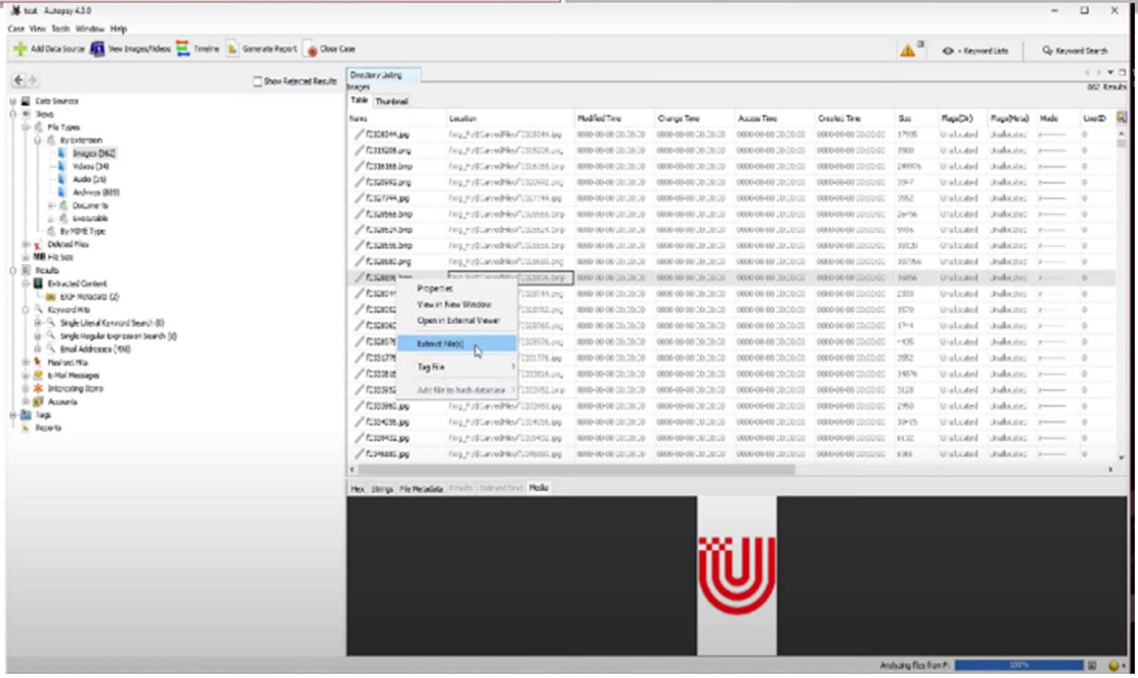
Select a local disk: Super Important Stuff (F:): 29.8 GB

Please select the input timezone: (GMT-6:00) America/Chicago

□ Ignore orphan files in FAT file systems

(faster results, although some data will not be searched)





Practical - 5

- **Aim:**
  To analyze network traffic using NetworkMiner to extract artifacts such as files, credentials, and session data from a captured PCAP file.

- **Tool/Application Used:**
  NetworkMiner

- **Theory:**
  NetworkMiner is a forensic analysis tool used for passive network traffic analysis. It allows investigators to extract data such as files, images, and credentials from captured network traffic (PCAP files). Unlike other tools, NetworkMiner focuses on extracting metadata and reconstructing transferred files instead of visualizing packets. This makes it ideal for post-incident analysis to investigate network breaches or anomalies.

- **Procedure:**

  1. **Launch NetworkMiner:**
     Open NetworkMiner on your system.

  2. **Load the PCAP File:**

     - Import the .pcap file by navigating to File > Open and selecting the captured network traffic file.

     - NetworkMiner will automatically process and parse the traffic data.

  3. **Analyze Hosts:**

     - Go to the Hosts tab to view a list of devices involved in the network communication.

     - Inspect IP addresses, hostnames, and MAC addresses for anomalies**.**

  4. **Analyze Credentials:**

     - Check the Credentials tab to identify usernames, passwords, and other authentication data transmitted over the network.

     - Note any plaintext credentials or anomalies.

  5. **Reconstruct Sessions:**

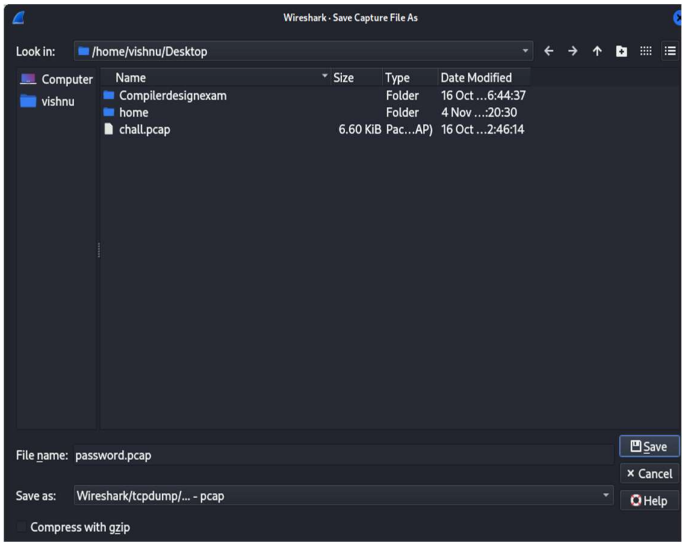
     - Use the Sessions tab to review individual network sessions for detailed traffic analysis.

     - Look for malicious activities, such as unauthorized file transfers or command and control communications.

  6. **Document Findings:**

     - Record any suspicious activities, extracted files, or sensitive data discovered during the analysis.

- **Output:**



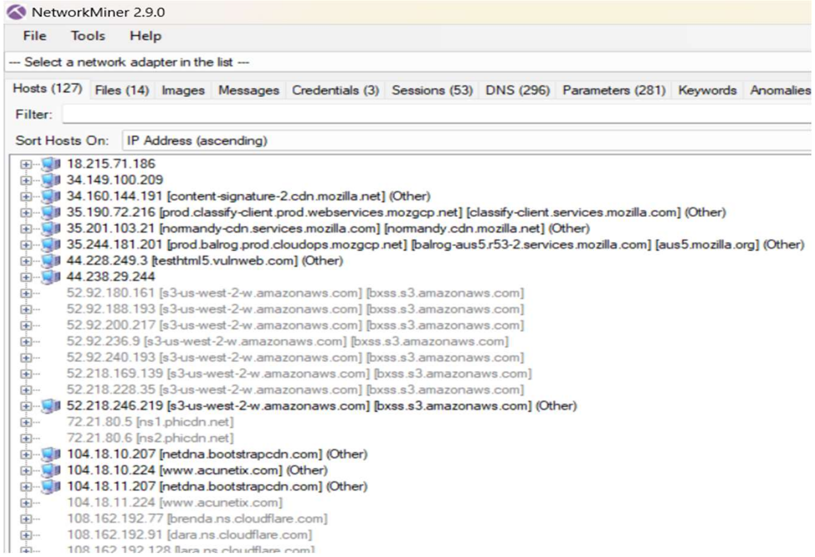
Look in: /home/vishnu/Desktop

Name	Size	Type	Date Modified
Compilerdesignexam		Folder	16 Oct …6:44:37
home		Folder	4 Nov …:20:30
chall.pcap	6.60 KiB	Pac…AP)	16 Oct …2:46:14

File name: password.pcap

Save as: Wireshark/tcpdump/... - pcap

Compress with gzip

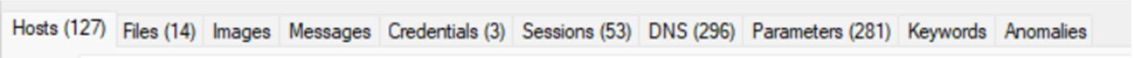


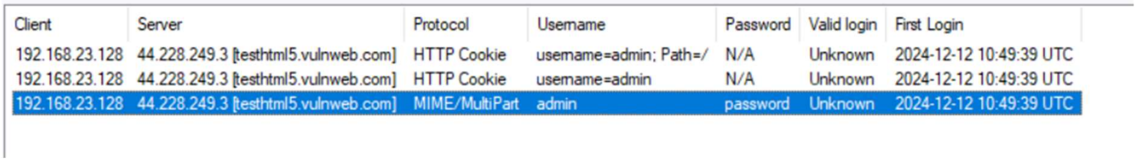
--- Select a network adapter in the list ---

Filter:

Sort Hosts On: IP Address (ascending)

```
18.215.71.186
34.149.100.209
34.160.144.191 [content-signature-2.cdn.mozilla.net] (Other)
35.190.72.216 [prod.classify-client.prod.webservices.mozgcp.net] [classify-client.services.mozilla.com] (Other)
35.201.103.21 [normandy-cdn.services.mozilla.com] [normandy.cdn.mozilla.net] (Other)
35.244.181.201 [prod.balrog.prod.cloudops.mozgcp.net] [balrog-aus5.r53-2.services.mozilla.com] [aus5.mozilla.org] (Other)
44.228.249.3 [testhtml5.vulnweb.com] (Other)
44.238.29.244
52.92.180.161 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.92.188.193 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.92.200.217 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.92.236.9 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.92.240.193 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.218.169.139 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.218.228.35 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com]
52.218.246.219 [s3-us-west-2-w.amazonaws.com] [bxss.s3.amazonaws.com] (Other)
72.21.80.5 [ns1.phicdn.net]
72.21.80.6 [ns2.phicdn.net]
104.18.10.207 [netdna.bootstrapcdn.com] (Other)
104.18.10.224 [www.acunetix.com] (Other)
104.18.11.207 [netdna.bootstrapcdn.com] (Other)
104.18.11.224 [www.acunetix.com]
108.162.192.77 [brenda.ns.cloudflare.com]
108.162.192.91 [dara.ns.cloudflare.com]
108.162.192.128 [lara.ns.cloudflare.com]
```





Client	Server	Protocol	Username	Password	Valid login	First Login
192.168.23.128	44.228.249.3 [testhtml5.vulnweb.com]	HTTP Cookie	usemame=admin; Path=/	N/A	Unknown	2024-12-12 10:49:39 UTC
192.168.23.128	44.228.249.3 [testhtml5.vulnweb.com]	HTTP Cookie	usemame=admin	N/A	Unknown	2024-12-12 10:49:39 UTC
192.168.23.128	44.228.249.3 [testhtml5.vulnweb.com]	MIME/MultiPart	admin	password	Unknown	2024-12-12 10:49:39 UTC

Practical - 6

- **Aim:**
  To use the netstat command to view information about incoming and outgoing network connections, routing tables, and interface statistics.

- **Tool/Application Used:**
  Command-line interface with the netstat utility (Linux/Windows).

- **Theory:**
  netstat (Network Statistics) is a command-line tool used to monitor and analyze network connections and performance. It provides insights into active connections, ports, protocols, routing tables, and interface statistics. This is particularly useful for diagnosing network-related issues, identifying unauthorized connections, or assessing system security.

- **Procedure:**

1. View Active Network Connections

- **Command:**

```
netstat
```

❖ Displays all active connections with details like protocol, local address, foreign address, and connection state.

2. Display Detailed Network Connections with Process ID (PID)

- **Command:**

```
netstat -a -n -o
```

❖ -a: Displays all connections and listening ports.

❖ -n: Displays addresses and port numbers in numerical form.

❖ -o: Shows the Process ID (PID) associated with each connection.

3. Filter Connections by Protocol (TCP or UDP)

- **Command (TCP):**

```
netstat -t
```

- **Command (UDP):**

```
netstat -u
```

4. Display Routing Table

- **Command:**

```
netstat -r
```

❖ Shows the system's routing table with destination networks, gateways, and interface information.

5. Display Network Interface Statistics

- **Command:**

  netstat -i

❖ Provides interface statistics like packets sent/received and errors.

6. Monitor Listening Ports and Applications

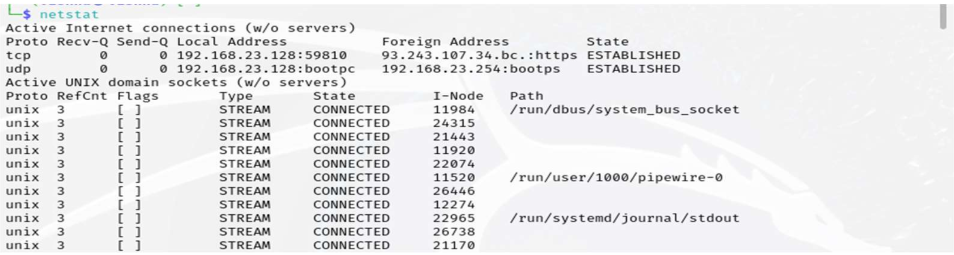
- **Command:**

  netstat -l

❖ Lists all listening ports and the applications using them.

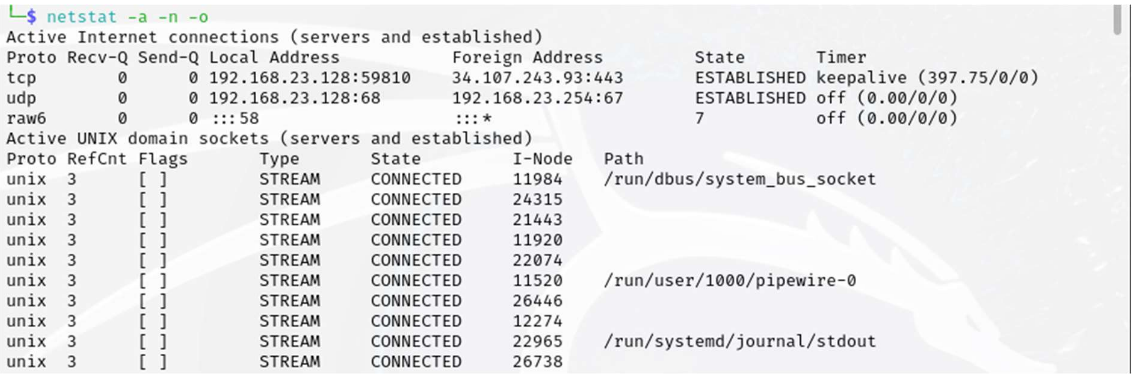
- **Output:**

1-



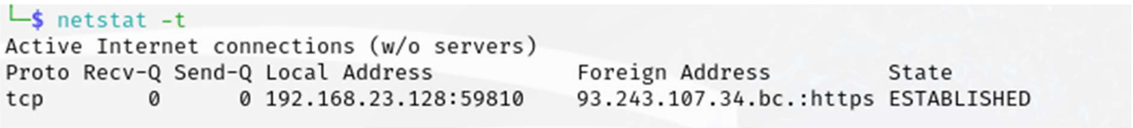
```
└─$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.23.128:59810    93.243.107.34.bc.:https ESTABLISHED
udp        0      0 192.168.23.128:bootpc   192.168.23.254:bootps   ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  3      [ ]         STREAM     CONNECTED     11984    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     24315
unix  3      [ ]         STREAM     CONNECTED     21443
unix  3      [ ]         STREAM     CONNECTED     11920
unix  3      [ ]         STREAM     CONNECTED     22074
unix  3      [ ]         STREAM     CONNECTED     11520    /run/user/1000/pipewire-0
unix  3      [ ]         STREAM     CONNECTED     26446
unix  3      [ ]         STREAM     CONNECTED     12274
unix  3      [ ]         STREAM     CONNECTED     22965    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     26738
unix  3      [ ]         STREAM     CONNECTED     21170
```

2-

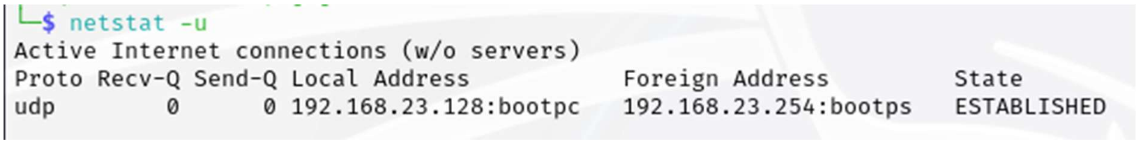


```
└─$ netstat -a -n -o
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       Timer
tcp        0      0 192.168.23.128:59810    34.107.243.93:443       ESTABLISHED keepalive (397.75/0/0)
udp        0      0 192.168.23.128:68       192.168.23.254:67       ESTABLISHED off (0.00/0/0)
raw6       0      0 :::58                   :::*                    7           off (0.00/0/0)
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  3      [ ]         STREAM     CONNECTED     11984    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     24315
unix  3      [ ]         STREAM     CONNECTED     21443
unix  3      [ ]         STREAM     CONNECTED     11920
unix  3      [ ]         STREAM     CONNECTED     22074
unix  3      [ ]         STREAM     CONNECTED     11520    /run/user/1000/pipewire-0
unix  3      [ ]         STREAM     CONNECTED     26446
unix  3      [ ]         STREAM     CONNECTED     12274
unix  3      [ ]         STREAM     CONNECTED     22965    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     26738
```

3-

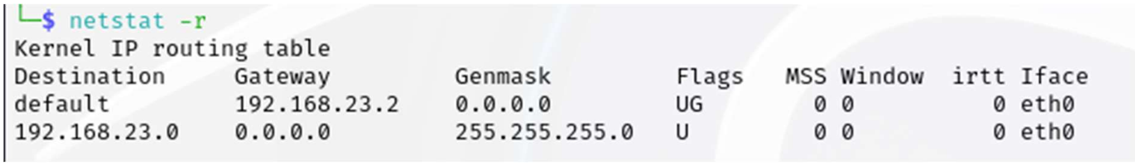


```
└─$ netstat -t
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.23.128:59810    93.243.107.34.bc.:https ESTABLISHED
```



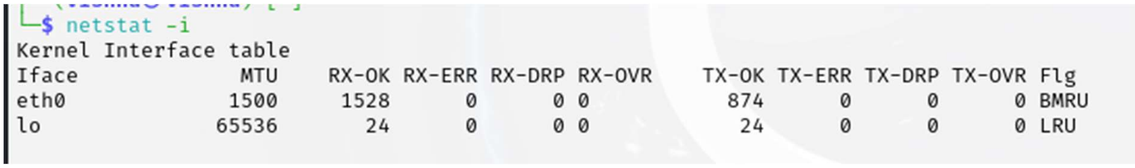
```
└─$ netstat -u
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp        0      0 192.168.23.128:bootpc   192.168.23.254:bootps   ESTABLISHED
```

4-



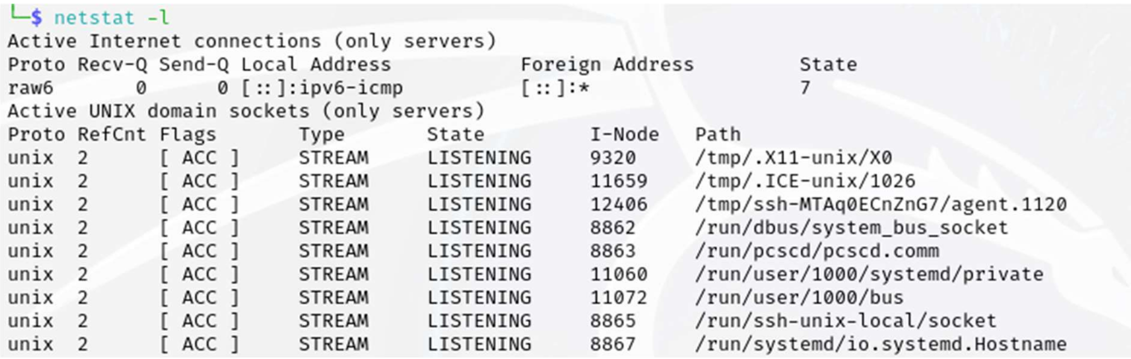
```
└─$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         192.168.23.2    0.0.0.0         UG        0 0          0 eth0
192.168.23.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

5-



```
└─$ netstat -i
Kernel Interface table
Iface             MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0             1500     1528      0      0 0           874      0      0      0 BMRU
lo              65536       24      0      0 0            24      0      0      0 LRU
```

6-



```
└─$ netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     9320     /tmp/.X11-unix/X0
unix  2      [ ACC ]     STREAM     LISTENING     11659    /tmp/.ICE-unix/1026
unix  2      [ ACC ]     STREAM     LISTENING     12406    /tmp/ssh-MTAq0ECnZnG7/agent.1120
unix  2      [ ACC ]     STREAM     LISTENING     8862     /run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     8863     /run/pcscd/pcscd.comm
unix  2      [ ACC ]     STREAM     LISTENING     11060    /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     11072    /run/user/1000/bus
unix  2      [ ACC ]     STREAM     LISTENING     8865     /run/ssh-unix-local/socket
unix  2      [ ACC ]     STREAM     LISTENING     8867     /run/systemd/io.systemd.Hostname
```

Practical - 7

- **Aim:**
  To monitor and manage active TCP/IP network connections on a Windows system using CurrPorts, identifying open ports, associated processes, and data transfer statistics.

- **Tool/Application Used:**
  CurrPorts (Windows)

- **Theory:**
  CurrPorts is a free network monitoring software for Windows that displays a list of all open TCP/IP and UDP ports on your local computer, including the processes associated with each connection. It provides real-time monitoring of network activity, allowing users to identify and manage network connections. It also shows data such as IP addresses, ports, protocols, and the data transfer rates for each connection. CurrPorts can be used to detect suspicious or unauthorized network connections, making it a useful tool for network administrators and security analysts.

- **Procedure:**

  1. Download and Install CurrPorts:

  - Visit the official CurrPorts website and download the software.
  - Install CurrPorts on your Windows system**.**

  2. Launch CurrPorts:

  - Open CurrPorts. The main window will display a list of all active network connections on your system.
  - It will show details such as:
    - Local Address/Port: The local machine's IP address and port.
    - Remote Address/Port: The remote machine's IP address and port.
    - Process Name and ID: The process associated with each network connection.
    - Protocol: TCP or UDP for each connection.
    - State: The state of the connection (e.g., Established, Listening, Time-Wait).

  3. Sort and Filter Connections:

  - You can sort the columns (e.g., by Process Name, State, or Protocol) to make it easier to analyze connections.
  - Filter connections by IP address, port number, or process to focus on specific network activity.

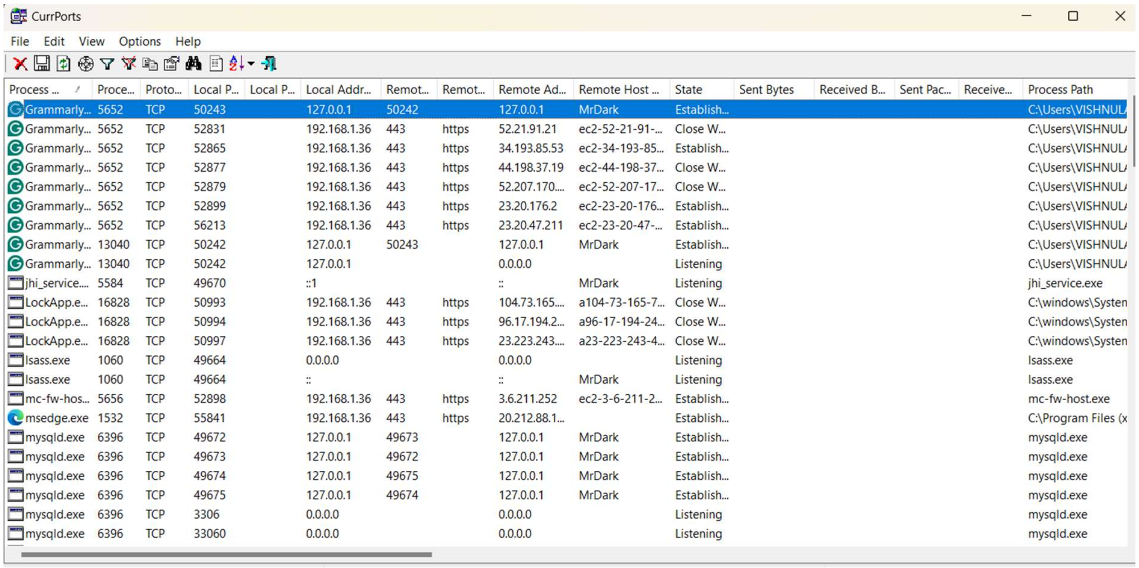
  4. Monitor Data Transfer Rates:

  - CurrPorts displays data transfer statistics for each connection, including the amount of data sent and received.
  - This can help in identifying high traffic connections or unexpected data usage.

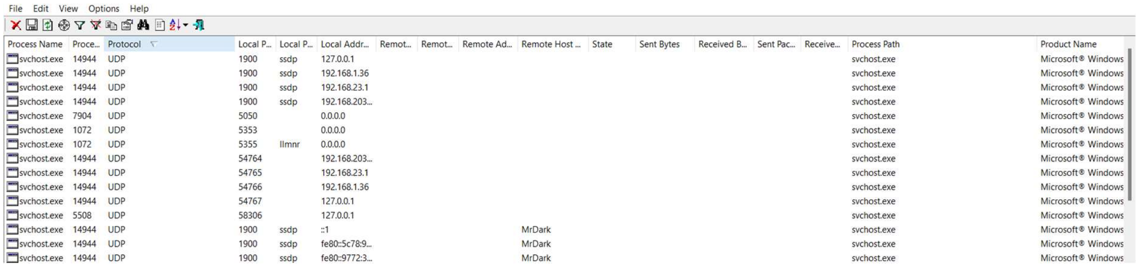
5. View Detailed Information About Processes:

- CurrPorts allows you to view detailed information about the process associated with each connection.
- Right-click on a process name and select "View Process Information" to get details such as the file path, version, and command line used to launch the process.

- **Output:**



Process ...	Proce...	Proto...	Local P...	Local P...	Local Addr...	Remot...	Remot...	Remote Ad...	Remote Host ...	State	Sent Bytes	Received B...	Sent Pac...	Receive...	Process Path
Grammarly...	5652	TCP	50243		127.0.0.1	50242		127.0.0.1	MrDark	Establish...					C:\Users\VISHNULA
Grammarly...	5652	TCP	52831		192.168.1.36	443	https	52.21.91.21	ec2-52-21-91-...	Close W...					C:\Users\VISHNULA
Grammarly...	5652	TCP	52865		192.168.1.36	443	https	34.193.85.53	ec2-34-193-85...	Establish...					C:\Users\VISHNULA
Grammarly...	5652	TCP	52877		192.168.1.36	443	https	44.198.37.19	ec2-44-198-37...	Close W...					C:\Users\VISHNULA
Grammarly...	5652	TCP	52879		192.168.1.36	443	https	52.207.170....	ec2-52-207-17...	Close W...					C:\Users\VISHNULA
Grammarly...	5652	TCP	52899		192.168.1.36	443	https	23.20.176.2	ec2-23-20-176...	Establish...					C:\Users\VISHNULA
Grammarly...	5652	TCP	56213		192.168.1.36	443	https	23.20.47.211	ec2-23-20-47-...	Establish...					C:\Users\VISHNULA
Grammarly...	13040	TCP	50242		127.0.0.1	50243		127.0.0.1	MrDark	Establish...					C:\Users\VISHNULA
Grammarly...	13040	TCP	50242		127.0.0.1			0.0.0.0		Listening					C:\Users\VISHNULA
jhi_service....	5584	TCP	49670		::1			::	MrDark	Listening					jhi_service.exe
LockApp.e...	16828	TCP	50993		192.168.1.36	443	https	104.73.165....	a104-73-165-7...	Close W...					C:\windows\Systen
LockApp.e...	16828	TCP	50994		192.168.1.36	443	https	96.17.194.2...	a96-17-194-24...	Close W...					C:\windows\Systen
LockApp.e...	16828	TCP	50997		192.168.1.36	443	https	23.223.243....	a23-223-243-4...	Close W...					C:\windows\Systen
lsass.exe	1060	TCP	49664		0.0.0.0			0.0.0.0		Listening					lsass.exe
lsass.exe	1060	TCP	49664		::			::	MrDark	Listening					lsass.exe
mc-fw-hos...	5656	TCP	52898		192.168.1.36	443	https	3.6.211.252	ec2-3-6-211-2...	Establish...					mc-fw-host.exe
msedge.exe	1532	TCP	55841		192.168.1.36	443	https	20.212.88.1...		Establish...					C:\Program Files (x
mysqld.exe	6396	TCP	49672		127.0.0.1	49673		127.0.0.1	MrDark	Establish...					mysqld.exe
mysqld.exe	6396	TCP	49673		127.0.0.1	49672		127.0.0.1	MrDark	Establish...					mysqld.exe
mysqld.exe	6396	TCP	49674		127.0.0.1	49675		127.0.0.1	MrDark	Establish...					mysqld.exe
mysqld.exe	6396	TCP	49675		127.0.0.1	49674		127.0.0.1	MrDark	Establish...					mysqld.exe
mysqld.exe	6396	TCP	3306		0.0.0.0			0.0.0.0		Listening					mysqld.exe
mysqld.exe	6396	TCP	33060		0.0.0.0			0.0.0.0		Listening					mysqld.exe



Process Name	Proce...	Protocol	Local P...	Local P...	Local Addr...	Remot...	Remot...	Remote Ad...	Remote Host ...	State	Sent Bytes	Received B...	Sent Pac...	Receive...	Process Path	Product Name
svchost.exe	14944	UDP	1900	ssdp	127.0.0.1										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	1900	ssdp	192.168.1.36										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	1900	ssdp	192.168.23.1										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	1900	ssdp	192.168.203...										svchost.exe	Microsoft® Windows
svchost.exe	7904	UDP	5050		0.0.0.0										svchost.exe	Microsoft® Windows
svchost.exe	1072	UDP	5353		0.0.0.0										svchost.exe	Microsoft® Windows
svchost.exe	1072	UDP	5355	llmnr	0.0.0.0										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	54764		192.168.203...										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	54765		192.168.23.1										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	54766		192.168.1.36										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	54767		127.0.0.1										svchost.exe	Microsoft® Windows
svchost.exe	5508	UDP	58306		127.0.0.1										svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	1900	ssdp	::1				MrDark						svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	1900	ssdp	fe80::5c78:9...				MrDark						svchost.exe	Microsoft® Windows
svchost.exe	14944	UDP	1900	ssdp	fe80::9772:3...				MrDark						svchost.exe	Microsoft® Windows

Practical - 8

- **Aim:**
  To monitor and manage network connections using TCPView, identifying active TCP/UDP connections and managing processes associated with them.

- **Tool/Application Used:**
  TCPView (Sysinternals Suite by Microsoft)

- **Theory:**
  TCPView is a graphical utility for Windows that provides a detailed overview of active TCP and UDP connections on a system. It displays information about endpoints, local and remote addresses, ports, connection states, and the processes associated with each connection. Unlike netstat, TCPView offers a real-time graphical interface, making it easier to monitor and manage network activities.

- **Procedure:**
  1. Download and Launch TCPView

    - Download TCPView from the official Sysinternals website.
    - Extract and run the Tcpview.exe file (administrator privileges may be required).

2. Observe Network Connections

  - The main TCPView window will display all active TCP and UDP connections, including:
    - Local Address and Port
    - Remote Address and Port
    - Connection State (e.g., Established, Listening, Time-Wait)
    - Process Name and PID

3. Highlight New Connections

  - Observe as new connections are highlighted in green, and closed connections are highlighted in red, allowing real-time monitoring of changes.

4. Sort and Filter Connections

  - Sort connections by clicking column headers (e.g., Process, Local Address, Remote Address).
  - Use filters to focus on specific applications or IP ranges.

5. Manage Connections

  - Right-click a connection to perform actions:
    - End Process: Terminates the associated process.
    - Close Connection: Closes the specific network connection without terminating the process.

6. Monitor Suspicious Activity

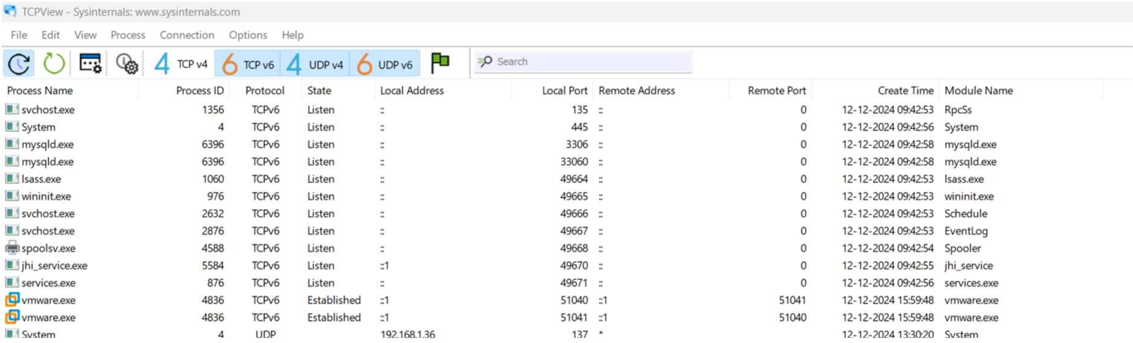
- Identify unusual remote addresses or ports.
- Check for unknown or suspicious processes associated with network activity.

7. Save Connection Logs

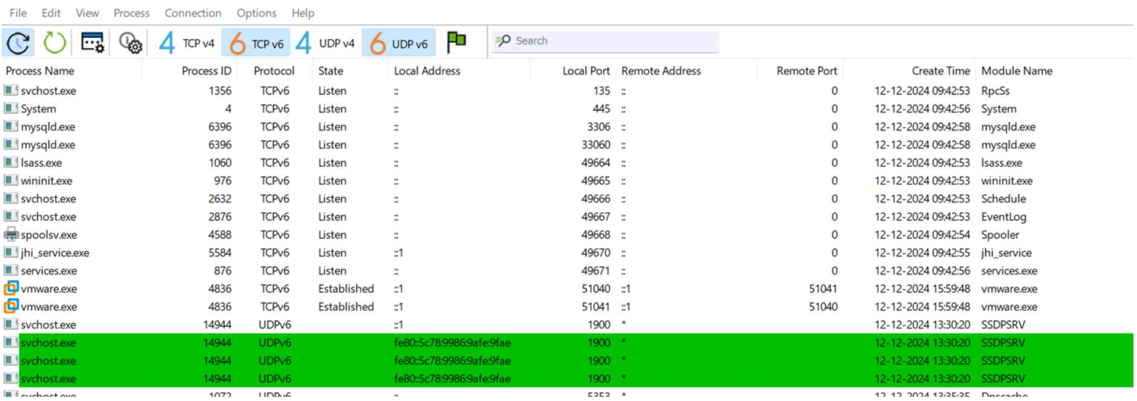
- Export the current view to a file for further analysis:

Go to File > Save As and choose a location to save the data.

- **Output:**



Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create Time	Module Name
svchost.exe	1356	TCPv6	Listen	::	135	::	0	12-12-2024 09:42:53	RpcSs
System	4	TCPv6	Listen	::	445	::	0	12-12-2024 09:42:56	System
mysqld.exe	6396	TCPv6	Listen	::	3306	::	0	12-12-2024 09:42:58	mysqld.exe
mysqld.exe	6396	TCPv6	Listen	::	33060	::	0	12-12-2024 09:42:58	mysqld.exe
lsass.exe	1060	TCPv6	Listen	::	49664	::	0	12-12-2024 09:42:53	lsass.exe
wininit.exe	976	TCPv6	Listen	::	49665	::	0	12-12-2024 09:42:53	wininit.exe
svchost.exe	2632	TCPv6	Listen	::	49666	::	0	12-12-2024 09:42:53	Schedule
svchost.exe	2876	TCPv6	Listen	::	49667	::	0	12-12-2024 09:42:53	EventLog
spoolsv.exe	4588	TCPv6	Listen	::	49668	::	0	12-12-2024 09:42:54	Spooler
jhi_service.exe	5584	TCPv6	Listen	::1	49670	::	0	12-12-2024 09:42:55	jhi_service
services.exe	876	TCPv6	Listen	::	49671	::	0	12-12-2024 09:42:56	services.exe
vmware.exe	4836	TCPv6	Established	::1	51040	::1	51041	12-12-2024 15:59:48	vmware.exe
vmware.exe	4836	TCPv6	Established	::1	51041	::1	51040	12-12-2024 15:59:48	vmware.exe
System	4	UDP		192.168.1.36	137	*		12-12-2024 13:30:20	System



Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create Time	Module Name
svchost.exe	1356	TCPv6	Listen	::	135	::	0	12-12-2024 09:42:53	RpcSs
System	4	TCPv6	Listen	::	445	::	0	12-12-2024 09:42:56	System
mysqld.exe	6396	TCPv6	Listen	::	3306	::	0	12-12-2024 09:42:58	mysqld.exe
mysqld.exe	6396	TCPv6	Listen	::	33060	::	0	12-12-2024 09:42:58	mysqld.exe
lsass.exe	1060	TCPv6	Listen	::	49664	::	0	12-12-2024 09:42:53	lsass.exe
wininit.exe	976	TCPv6	Listen	::	49665	::	0	12-12-2024 09:42:53	wininit.exe
svchost.exe	2632	TCPv6	Listen	::	49666	::	0	12-12-2024 09:42:53	Schedule
svchost.exe	2876	TCPv6	Listen	::	49667	::	0	12-12-2024 09:42:53	EventLog
spoolsv.exe	4588	TCPv6	Listen	::	49668	::	0	12-12-2024 09:42:54	Spooler
jhi_service.exe	5584	TCPv6	Listen	::1	49670	::	0	12-12-2024 09:42:55	jhi_service
services.exe	876	TCPv6	Listen	::	49671	::	0	12-12-2024 09:42:56	services.exe
vmware.exe	4836	TCPv6	Established	::1	51040	::1	51041	12-12-2024 15:59:48	vmware.exe
vmware.exe	4836	TCPv6	Established	::1	51041	::1	51040	12-12-2024 15:59:48	vmware.exe
svchost.exe	14944	UDPv6		::1	1900	*		12-12-2024 13:30:20	SSDPSRV
svchost.exe	14944	UDPv6		fe80::5c78:9986:9afe:9fae	1900	*		12-12-2024 13:30:20	SSDPSRV
svchost.exe	14944	UDPv6		fe80::5c78:9986:9afe:9fae	1900	*		12-12-2024 13:30:20	SSDPSRV
svchost.exe	14944	UDPv6		fe80::5c78:9986:9afe:9fae	1900	*		12-12-2024 13:30:20	SSDPSRV
svchost.exe	1072	UDPv6		::	5353	*		12-12-2024 13:35:35	Dnscache

Practical - 9

- **Aim:**
  To scan a host using Nmap to gather information about open ports, services, and system details, and understand the results.
- **Tool/Application Used:**
  Nmap (Network Mapper)
- **Theory:**
  Nmap is an open-source network scanning tool used to discover hosts, services, and vulnerabilities on a network. It can be used to scan a single host or a range of IP addresses. Nmap works by sending specially crafted packets to the target and analyzing the responses to determine the state of open ports and services. Common results from Nmap scans include open ports, service versions, and potential security issues.
- **Procedure:**

1. Basic Host Scan:
To perform a basic scan of a target host (replace target_ip with the IP address of the target):

nmap 192.168.23.1

This command scans the target for common ports and provides basic information on open ports.

2. Scan Specific Ports:
To scan specific ports (e.g., ports 22, 80, 443):

nmap -p 22,80,443 192.168.23.1

This command scans only the specified ports and shows whether they are open or closed.

3. Service Version Detection:
To detect versions of services running on open ports:

nmap -sV 192.168.23.1

This will attempt to identify the version of the services running on open ports (e.g., Apache 2.4.29 or OpenSSH 7.6).

4. Operating System Detection:
To attempt to detect the target system's operating system:

nmap -O 192.168.23.1

This will try to identify the OS by analyzing TCP/IP stack characteristics.

5. Aggressive Scan:
An aggressive scan scans for open ports, detects services, performs OS detection, and runs scripts to detect vulnerabilities:

nmap -A 192.168.23.1

This comprehensive scan can take longer and provide more detailed results, including possible vulnerabilities.

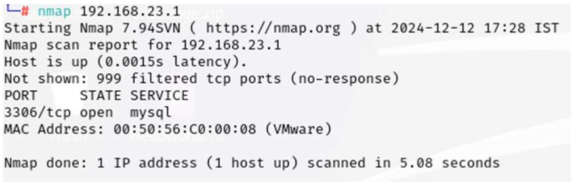
6. Scan Multiple Hosts:
To scan a range of IPs (e.g., 192.168.1.1 to 192.168.1.10):

nmap 192.168.1.1-10

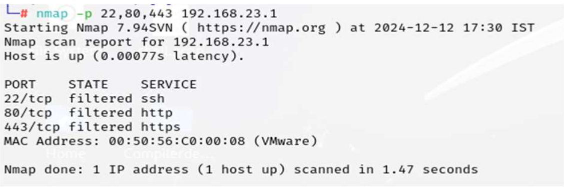
7. Scan Using UDP:
To scan for open UDP ports (e.g., port 161 for SNMP):

nmap -sU -p 161 192.168.23.1

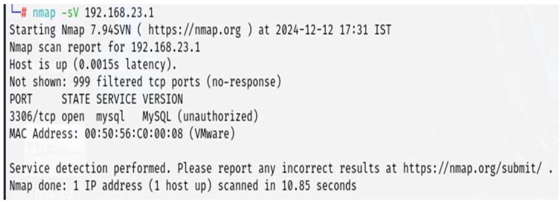
Output



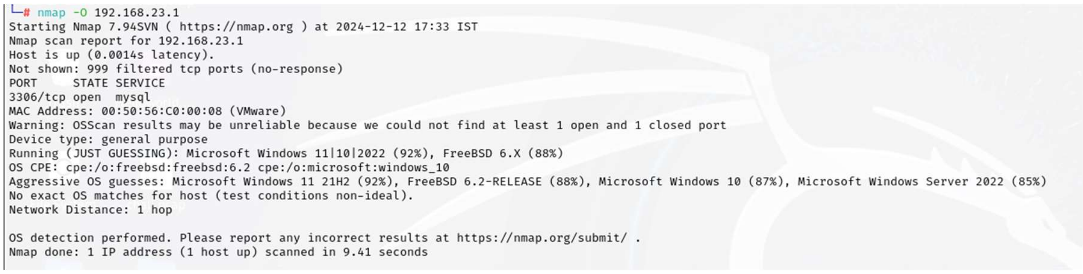
```
└─# nmap 192.168.23.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:28 IST
Nmap scan report for 192.168.23.1
Host is up (0.0015s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT     STATE SERVICE
3306/tcp open  mysql
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 5.08 seconds
```



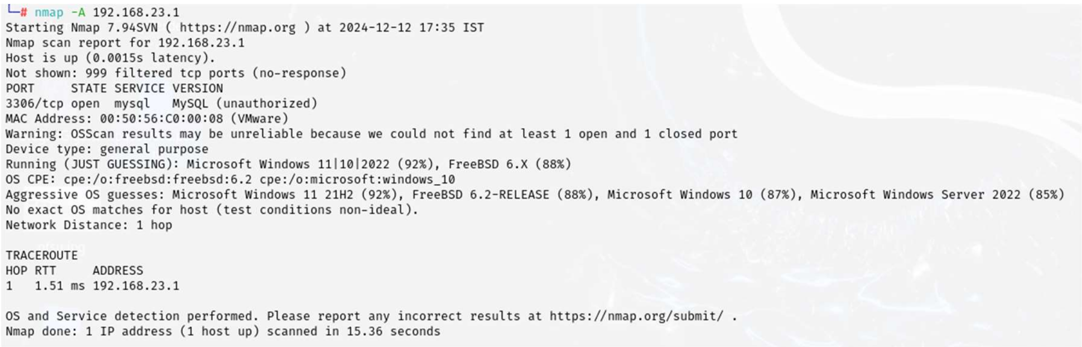
```
└─# nmap -p 22,80,443 192.168.23.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:30 IST
Nmap scan report for 192.168.23.1
Host is up (0.00077s latency).

PORT    STATE    SERVICE
22/tcp  filtered ssh
80/tcp  filtered http
443/tcp filtered https
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.47 seconds
```



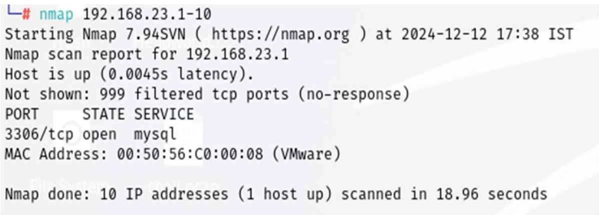
```
└─# nmap -sV 192.168.23.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:31 IST
Nmap scan report for 192.168.23.1
Host is up (0.0015s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT     STATE SERVICE VERSION
3306/tcp open  mysql   MySQL (unauthorized)
MAC Address: 00:50:56:C0:00:08 (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.85 seconds
```



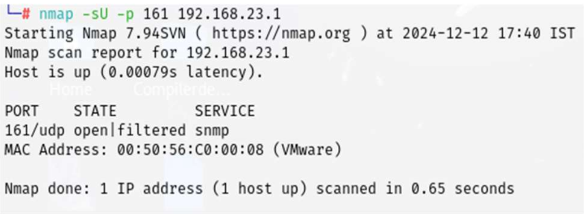
```
└─# nmap -O 192.168.23.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:33 IST
Nmap scan report for 192.168.23.1
Host is up (0.0014s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT     STATE SERVICE
3306/tcp open  mysql
MAC Address: 00:50:56:C0:00:08 (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 11|10|2022 (92%), FreeBSD 6.X (88%)
OS CPE: cpe:/o:freebsd:freebsd:6.2 cpe:/o:microsoft:windows_10
Aggressive OS guesses: Microsoft Windows 11 21H2 (92%), FreeBSD 6.2-RELEASE (88%), Microsoft Windows 10 (87%), Microsoft Windows Server 2022 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.41 seconds
```



```
└─# nmap -A 192.168.23.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:35 IST
Nmap scan report for 192.168.23.1
Host is up (0.0015s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT     STATE SERVICE VERSION
3306/tcp open  mysql   MySQL (unauthorized)
MAC Address: 00:50:56:C0:00:08 (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 11|10|2022 (92%), FreeBSD 6.X (88%)
OS CPE: cpe:/o:freebsd:freebsd:6.2 cpe:/o:microsoft:windows_10
Aggressive OS guesses: Microsoft Windows 11 21H2 (92%), FreeBSD 6.2-RELEASE (88%), Microsoft Windows 10 (87%), Microsoft Windows Server 2022 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   1.51 ms 192.168.23.1

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.36 seconds
```



```
└─# nmap 192.168.23.1-10
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:38 IST
Nmap scan report for 192.168.23.1
Host is up (0.0045s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT     STATE SERVICE
3306/tcp open  mysql
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap done: 10 IP addresses (1 host up) scanned in 18.96 seconds
```



```
└─# nmap -sU -p 161 192.168.23.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 17:40 IST
Nmap scan report for 192.168.23.1
Host is up (0.00079s latency).

PORT    STATE         SERVICE
161/udp open|filtered snmp
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.65 seconds
```