Wireshark: Traffic Analysis

1 - Nmap Scans:

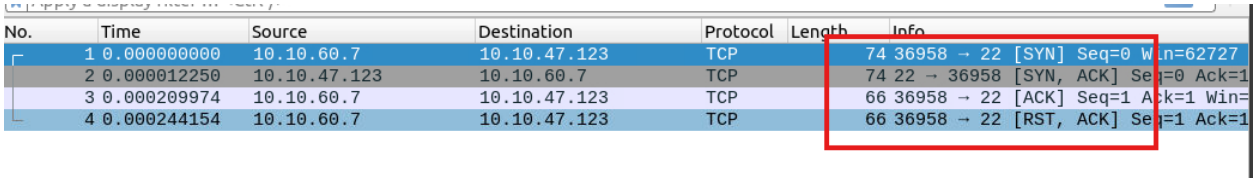
This section will cover identifying the most common Nmap scan types.

- TCP connect scans
- SYN scans
- UDP scans

- TCP Connect Scans:

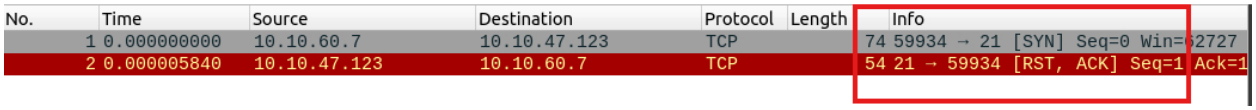
- **Relies on the three-way handshake**
- **Usually conducted with `nmap -sT` command.**
- **Used by non-privileged .**
- **Usually has a windows size larger than 1024 bytes as the request expects some data due to the nature of the protocol.**

Open TCP Port:



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.10.60.7	10.10.47.123	TCP	74	36958 → 22 [SYN] Seq=0 Win=62727
2	0.000012250	10.10.47.123	10.10.60.7	TCP	74	22 → 36958 [SYN, ACK] Seq=0 Ack=1
3	0.000209974	10.10.60.7	10.10.47.123	TCP	66	36958 → 22 [ACK] Seq=1 Ack=1 Win=
4	0.000244154	10.10.60.7	10.10.47.123	TCP	66	36958 → 22 [RST, ACK] Seq=1 Ack=1

Closed TCP port :

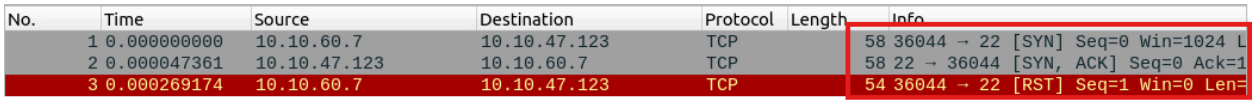


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.10.60.7	10.10.47.123	TCP	74	59934 → 21 [SYN] Seq=0 Win=62727
2	0.000005840	10.10.47.123	10.10.60.7	TCP	54	21 → 59934 [RST, ACK] Seq=1 Ack=1

- SYN Scans:

- Doesn't rely on the three-way handshake
- Usually conducted with `nmap -sS` command.
- Used by privileged users.
- Usually have a size less than or equal to 1024 bytes as the request is not finished and it doesn't expect to receive data.

Open TCP port (SYN):



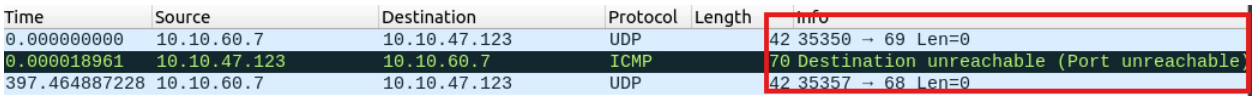
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.10.60.7	10.10.47.123	TCP	58	36044 → 22 [SYN] Seq=0 Win=1024 L
2	0.000047361	10.10.47.123	10.10.60.7	TCP	58	22 → 36044 [SYN, ACK] Seq=0 Ack=1
3	0.000269174	10.10.60.7	10.10.47.123	TCP	54	36044 → 22 [RST] Seq=1 Win=0 Len=

Closed TCP port (SYN):

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.10.60.7	10.10.47.123	TCP	58	36044 → 21 [SYN] Seq=0 Win=1024 L
2	0.000007060	10.10.47.123	10.10.60.7	TCP	54	21 → 36044 [RST, ACK] Seq=1 Ack=1

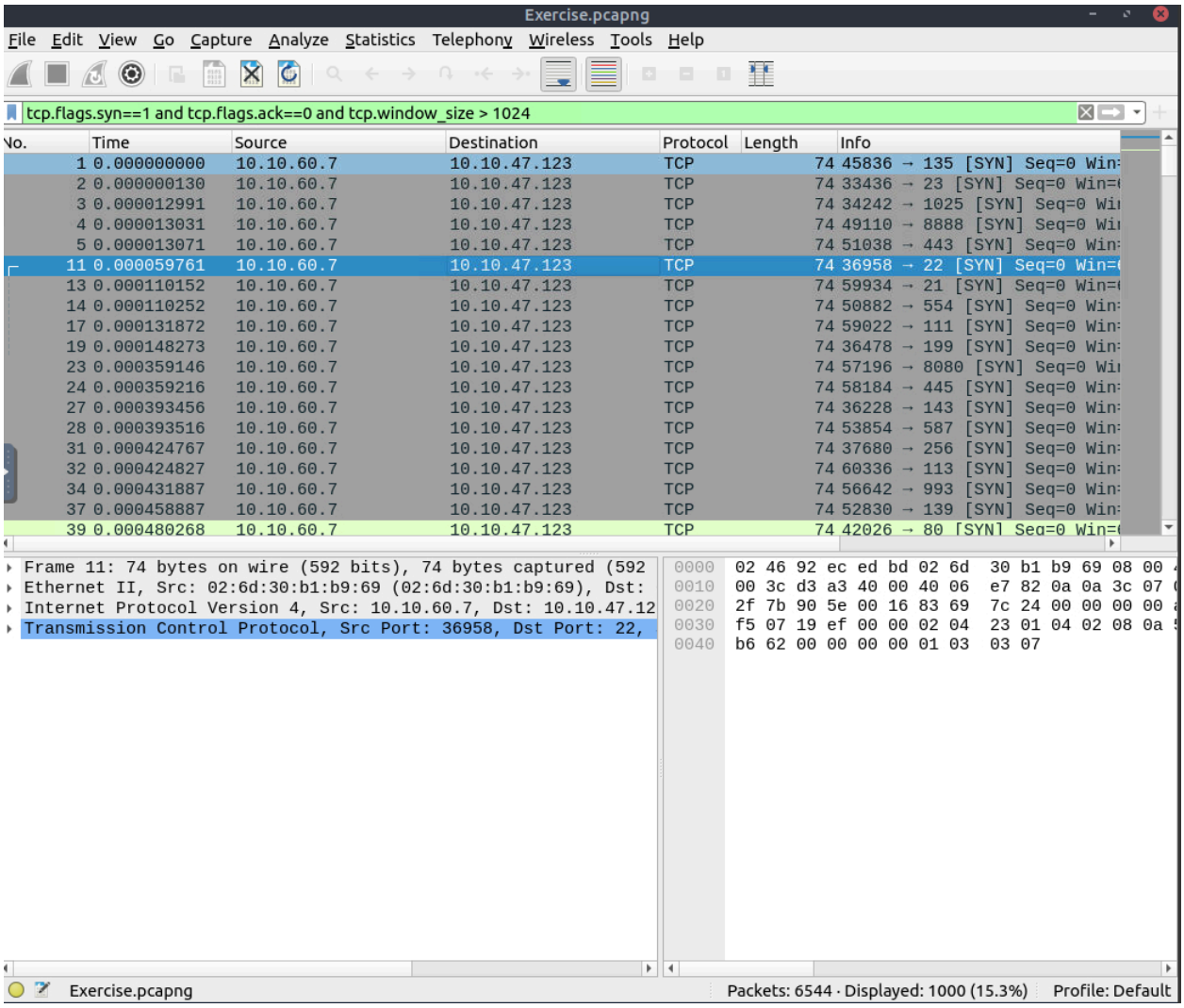
- # UDP Scans:
  - **Doesn't require a handshake process**
  - **No prompt for open ports**
  - **ICMP error message for close ports**
  - **Usually conducted with `nmap -sU` command.**

Closed (port no 69) and open (port no 68) UDP ports:



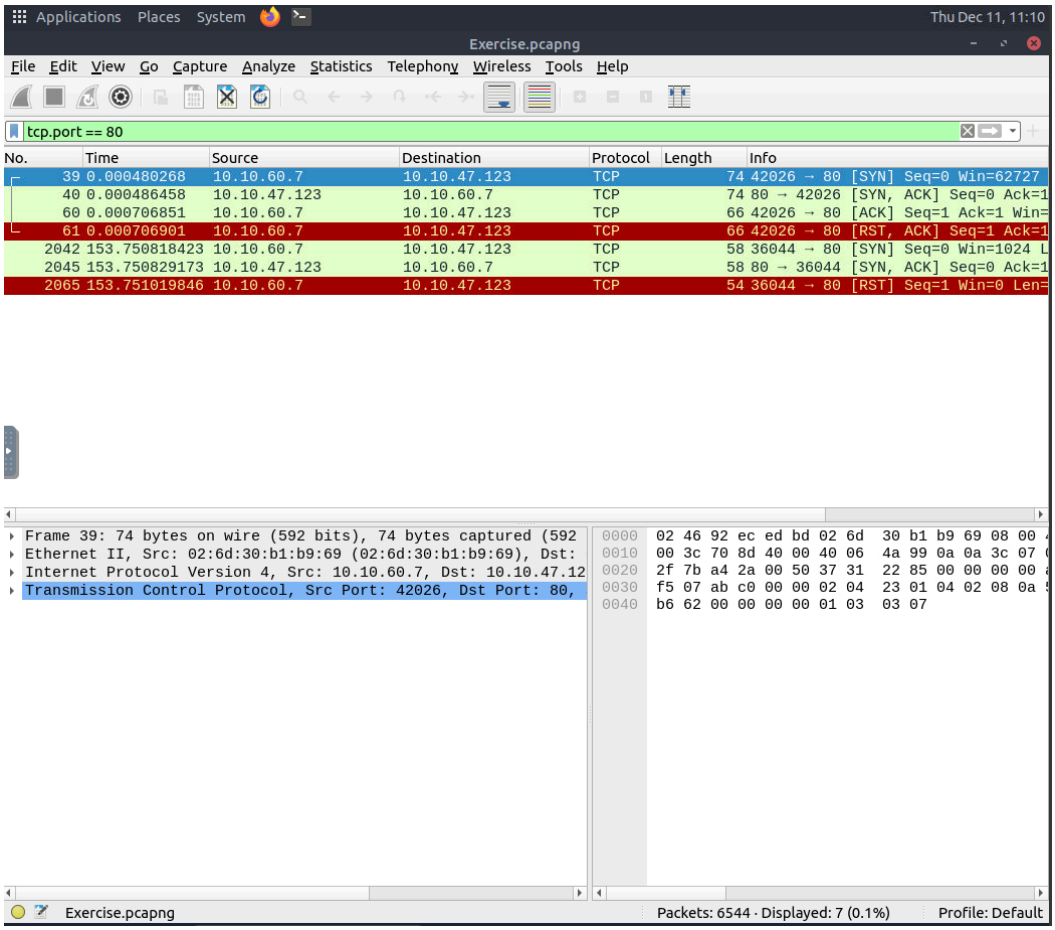
Time	Source	Destination	Protocol	Length	Info
0.000000000	10.10.60.7	10.10.47.123	UDP		42 35350 → 69 Len=0
0.000018961	10.10.47.123	10.10.60.7	ICMP		70 Destination unreachable (Port unreachable)
397.464887228	10.10.60.7	10.10.47.123	UDP		42 35357 → 68 Len=0

1-What is the total number of the "TCP Connect" scans?The answer : 1000



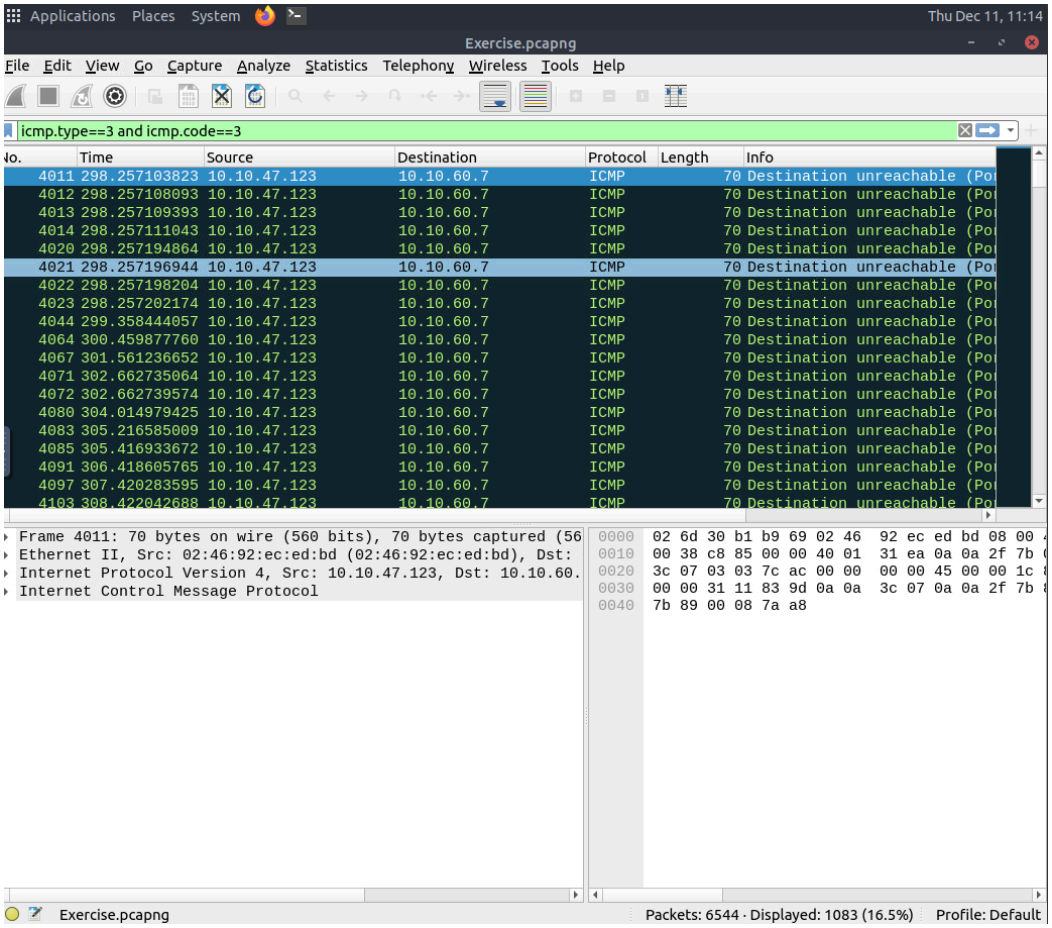
2-Which scan type is used to scan the TCP port 80?

The answer : TCP Connect



3-How many "UDP close port" messages are there?

The answer :1083

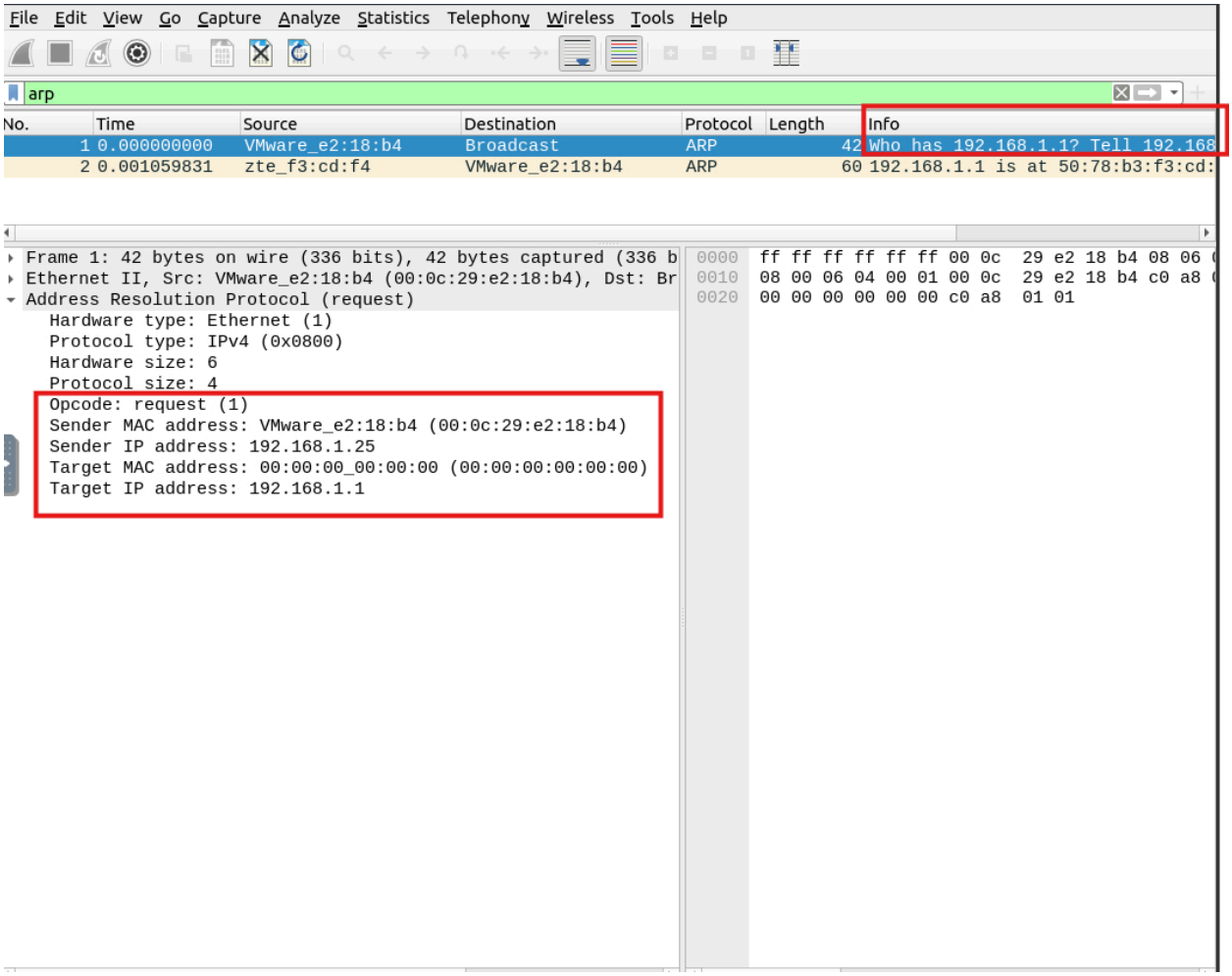


2-ARP Poisoning/Spoofing (A.K.A. Man In The Middle Attack):

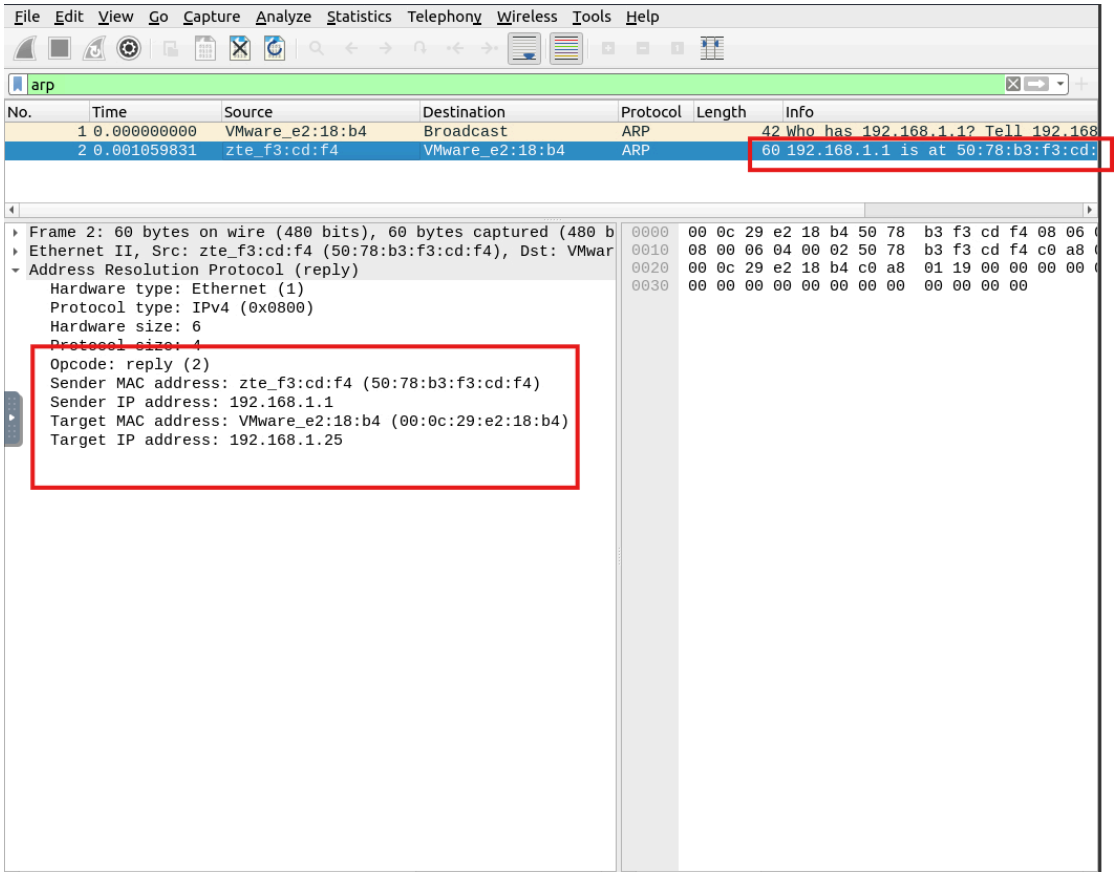
- Works on the local network
- Enables the communication between MAC addresses
- Not a secure protocol
- Not a routable protocol
- It doesn't have an authentication function

- Common patterns are request & response, announcement and gratuitous packets.

ARP Request:



ARP Reply:



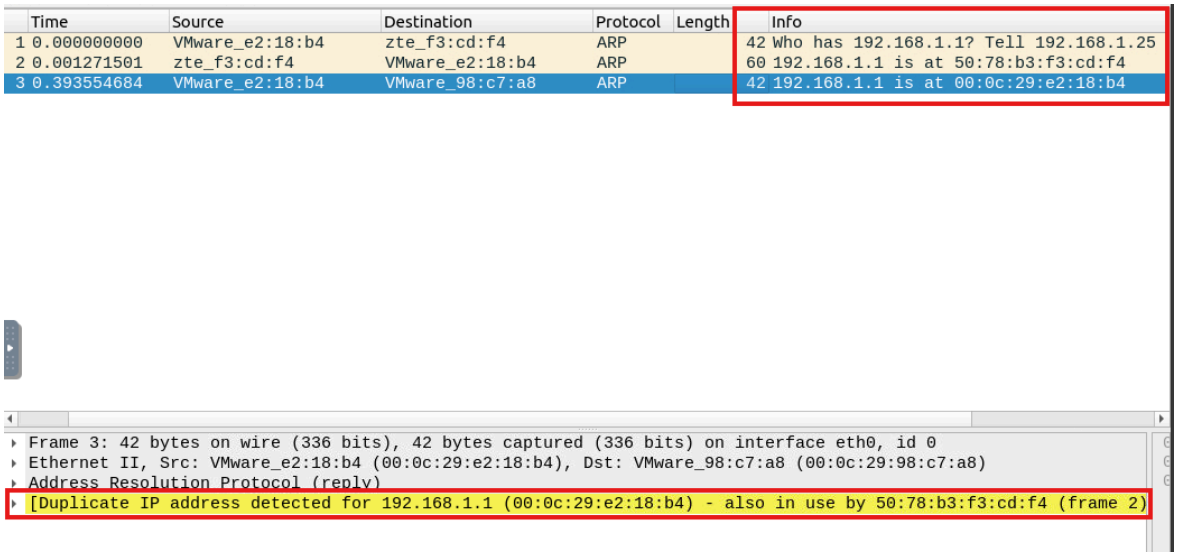
Filters used to detect spoofing:

- **arp.dst.hw_mac == 00:00:00:00:00:00:**Attackers sometimes use a zero-MAC destination in crafted ARP packets during spoofing or poisoning attempts, because it can trick devices into updating ARP tables.

- **arp.duplicate-address-detected or arp.duplicate-address-frame:** This filter shows ARP packets where Wireshark has noticed two devices claiming the same IP address.
- **((arp) && (arp.opcode == 1)) && (arp.src.hw_mac == target-mac-address):**It shows only ARP requests (opcode 1) that come from a specific MAC address.

  Here, knowing the network architecture and inspecting the traffic for a specific time frame can help detect the anomaly. As an analyst, you should take notes of your findings before going further. This will help you be organised and make it easier to correlate the further findings. Look at the given picture; there is a conflict; the MAC address that ends

  with "b4" crafted an ARP request with the "192.168.1.25" IP address, then claimed to have the "192.168.1.1" IP address.



	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	VMware_e2:18:b4	zte_f3:cd:f4	ARP	42	Who has 192.168.1.1? Tell 192.168.1.25
2	0.001271501	zte_f3:cd:f4	VMware_e2:18:b4	ARP	60	192.168.1.1 is at 50:78:b3:f3:cd:f4
3	0.393554684	VMware_e2:18:b4	VMware_98:c7:a8	ARP	42	192.168.1.1 is at 00:0c:29:e2:18:b4

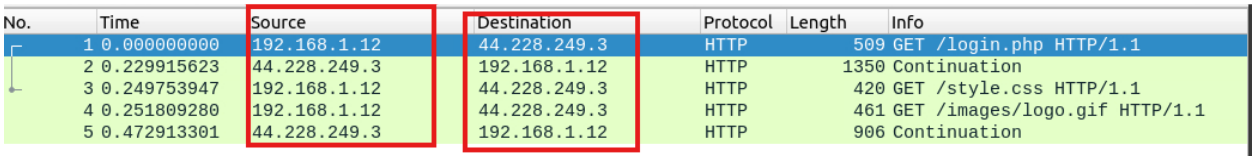
```
Frame 3: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface eth0, id 0
Ethernet II, Src: VMware_e2:18:b4 (00:0c:29:e2:18:b4), Dst: VMware_98:c7:a8 (00:0c:29:98:c7:a8)
Address Resolution Protocol (reply)
[Duplicate IP address detected for 192.168.1.1 (00:0c:29:e2:18:b4) - also in use by 50:78:b3:f3:cd:f4 (frame 2)]
```

At this point, it is evident that there is an anomaly. A security analyst cannot ignore a flood of ARP requests. This could be malicious activity, scan or network problems. There is a new anomaly; the MAC address that ends with "b4" crafted multiple ARP requests with the "192.168.1.25" IP address. Let's focus on the source of this anomaly and extend the taken notes.

	Source	Destination	Protocol	Length	Info
0000000	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.25
1059831	zte_f3:cd:f4	VMware_e2:18:b4	ARP	60	192.168.1.1 is at 50:78:b3:f3:cd:f4
9490253	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.37? Tell 192.168.1.25
9876839	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.158? Tell 192.168.1.25
1275021	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.212? Tell 192.168.1.25
1848453	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.176? Tell 192.168.1.25
2746298	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.73? Tell 192.168.1.25
3388601	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.216? Tell 192.168.1.25
3905794	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.181? Tell 192.168.1.25
4401792	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.217? Tell 192.168.1.25
5003040	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.173? Tell 192.168.1.25
5417559	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.136? Tell 192.168.1.25
5638938	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.132? Tell 192.168.1.25
5920898	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.130? Tell 192.168.1.25
6708415	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.1.25
7294383	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.232? Tell 192.168.1.25
7926474	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.162? Tell 192.168.1.25
8416850	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.109? Tell 192.168.1.25
8936116	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.253? Tell 192.168.1.25
9453050	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.169? Tell 192.168.1.25
9958386	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.145? Tell 192.168.1.25
0437055	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.127? Tell 192.168.1.25
0961982	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.227? Tell 192.168.1.25
1445342	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.163? Tell 192.168.1.25
1972828	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.199? Tell 192.168.1.25
2453951	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.208? Tell 192.168.1.25
2965087	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.226? Tell 192.168.1.25
3128368	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.166? Tell 192.168.1.25
3243915	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.154? Tell 192.168.1.25
3634125	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.113? Tell 192.168.1.25
4149337	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.237? Tell 192.168.1.25
4340485	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.159? Tell 192.168.1.25
4788873	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.119? Tell 192.168.1.25
5021487	VMware_e2:18:b4	Broadcast	ARP	42	Who has 192.168.1.91? Tell 192.168.1.25

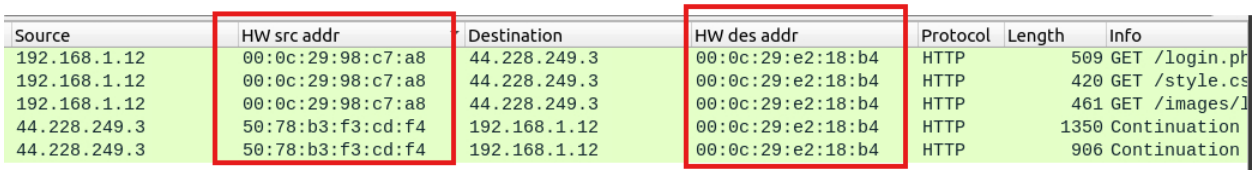
Notes	Detection Notes	Findings
Possible IP address match.	1 IP address announced from a MAC address.	• MAC: 00:0c:29:e2:18:b4 • IP: 192.168.1.25
Possible ARP spoofing attempt.	2 MAC addresses claimed the same IP address (192.168.1.1). The " 192.168.1.1" IP address is a possible gateway address.	• MAC1: 50:78:b3:f3:cd:f4 • MAC 2: 00:0c:29:e2:18:b4
Possible ARP spoofing attempt.	The MAC address that ends with "b4" claims to have a different/new IP address.	• MAC: 00:0c:29:e2:18:b4 • IP: 192.168.1.1
Possible ARP flooding attempt.	The MAC address that ends with "b4" crafted multiple ARP requests against a range of IP addresses.	• MAC: 00:0c:29:e2:18:b4 • IP: 192.168.1.xxx

There is HTTP traffic, and everything looks normal at the IP level, so there is no linked information with our previous findings. Let's add the MAC addresses as columns in the packet list pane to reveal the communication behind the IP addresses.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.12	44.228.249.3	HTTP	509	GET /login.php HTTP/1.1
2	0.229915623	44.228.249.3	192.168.1.12	HTTP	1350	Continuation
3	0.249753947	192.168.1.12	44.228.249.3	HTTP	420	GET /style.css HTTP/1.1
4	0.251809280	192.168.1.12	44.228.249.3	HTTP	461	GET /images/logo.gif HTTP/1.1
5	0.472913301	44.228.249.3	192.168.1.12	HTTP	906	Continuation

One more anomaly! The MAC address that ends with "b4" is the destination of all HTTP packets! It is evident that there is a MITM attack, and the attacker is the host with the MAC address that ends with "b4". All traffic linked to "192.168.1.12" IP addresses is forwarded to the malicious host. Let's summarise the findings before concluding the investigation.

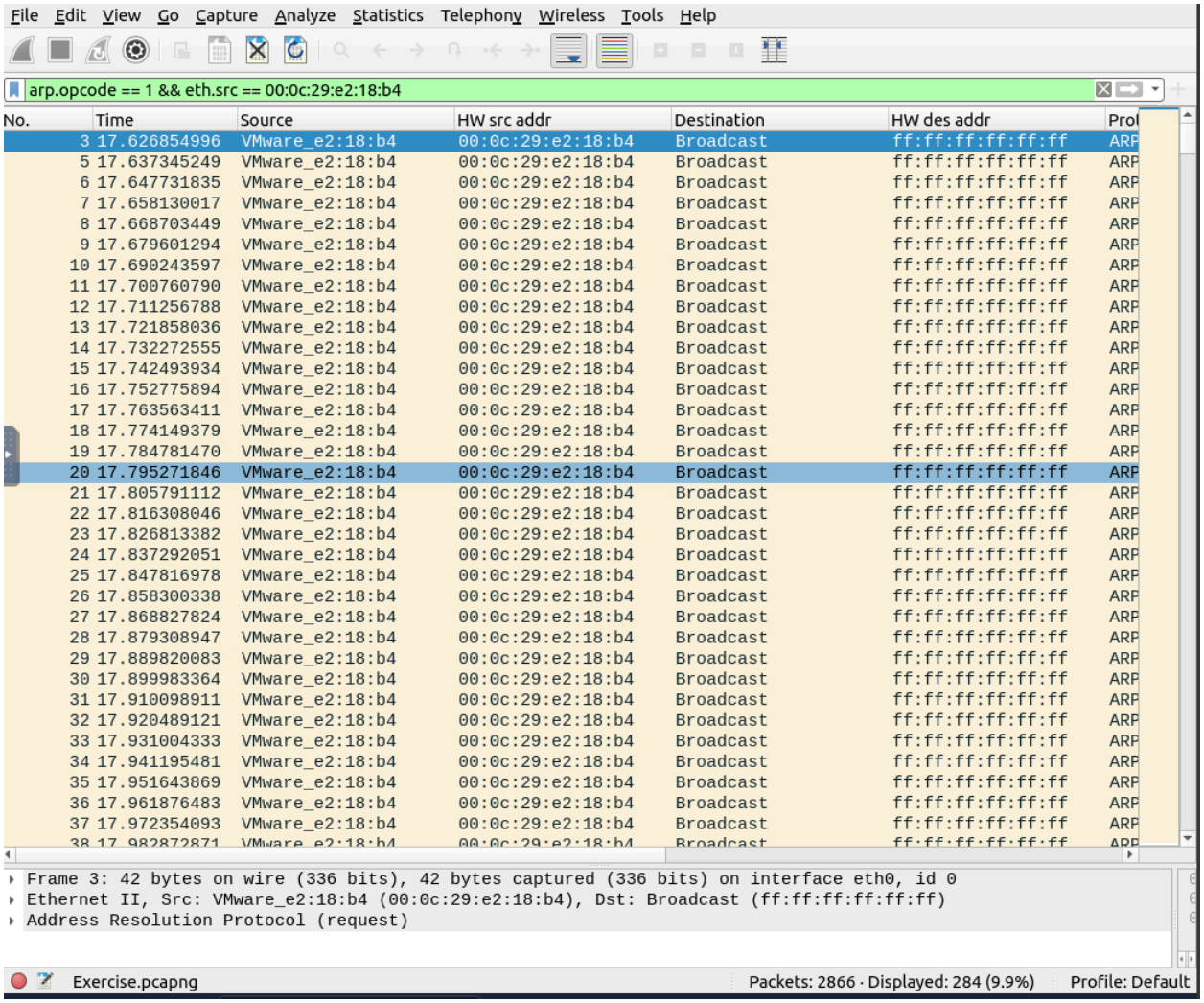


Source	HW src addr	Destination	HW des addr	Protocol	Length	Info
192.168.1.12	00:0c:29:98:c7:a8	44.228.249.3	00:0c:29:e2:18:b4	HTTP	509	GET /login.ph
192.168.1.12	00:0c:29:98:c7:a8	44.228.249.3	00:0c:29:e2:18:b4	HTTP	420	GET /style.cs
192.168.1.12	00:0c:29:98:c7:a8	44.228.249.3	00:0c:29:e2:18:b4	HTTP	461	GET /images/l
44.228.249.3	50:78:b3:f3:cd:f4	192.168.1.12	00:0c:29:e2:18:b4	HTTP	1350	Continuation
44.228.249.3	50:78:b3:f3:cd:f4	192.168.1.12	00:0c:29:e2:18:b4	HTTP	906	Continuation

Notes	Detection Notes	Findings
IP to MAC matches.	3 IP to MAC address matches.	• MAC: 00:0c:29:e2:18:b4 = IP: 192.168.1.25 • MAC: 50:78:b3:f3:cd:f4 = IP: 192.1681.1 • MAC: 00:0c:29:98:c7:a8 = IP: 192.168.1.12
Attacker	The attacker created noise with ARP packets.	• MAC: 00:0c:29:e2:18:b4 = IP: 192.168.1.25
Router/gateway	Gateway address.	• MAC: 50:78:b3:f3:cd:f4 = IP: 192.1681.1
Victim	The attacker sniffed all traffic of the victim.	• MAC: 50:78:b3:f3:cd:f4 = IP: 192.1681.12

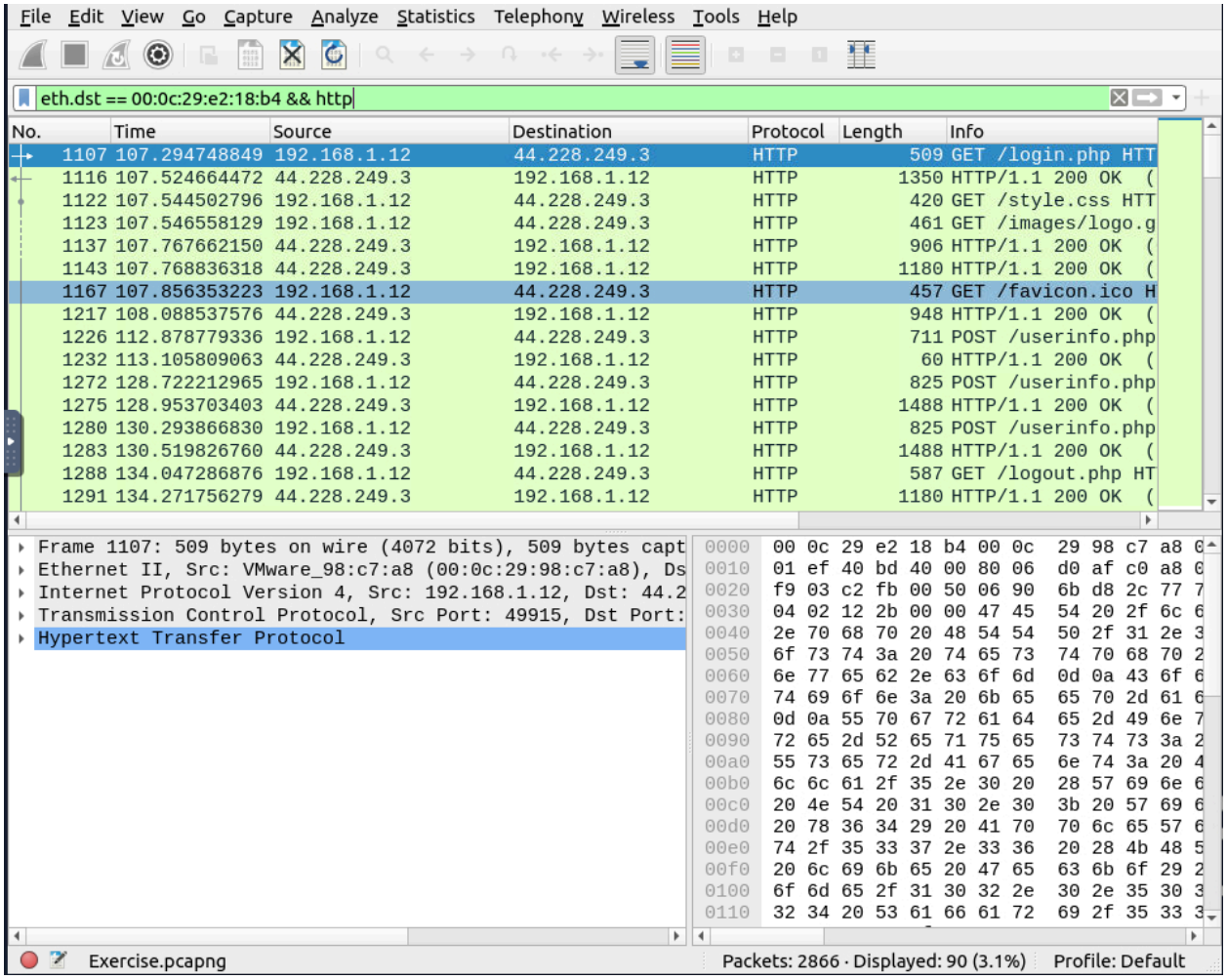
1-What is the number of ARP requests crafted by the attacker?

The answer :284



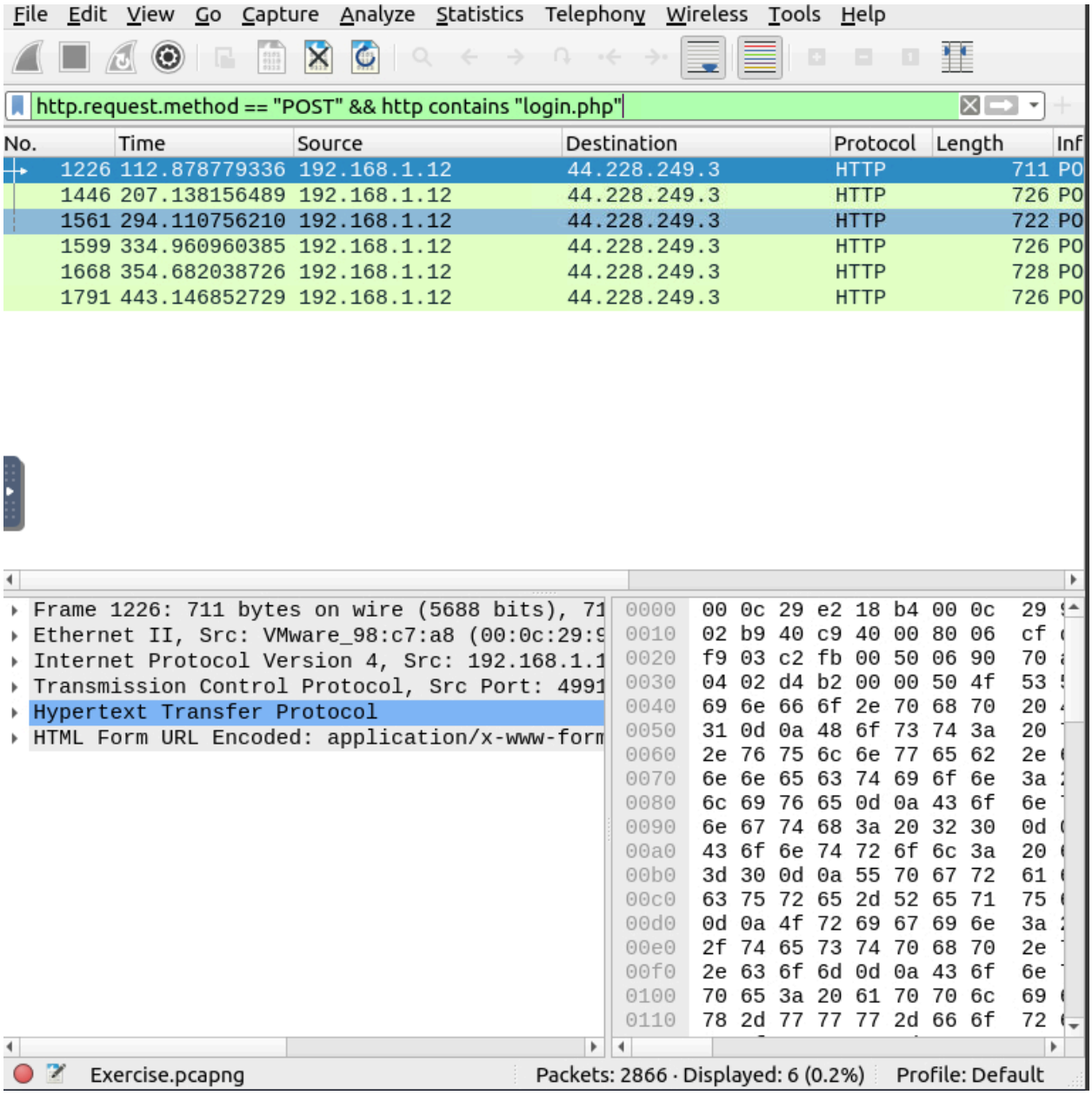
2-What is the number of sniffed username&password entries?

The answer :90

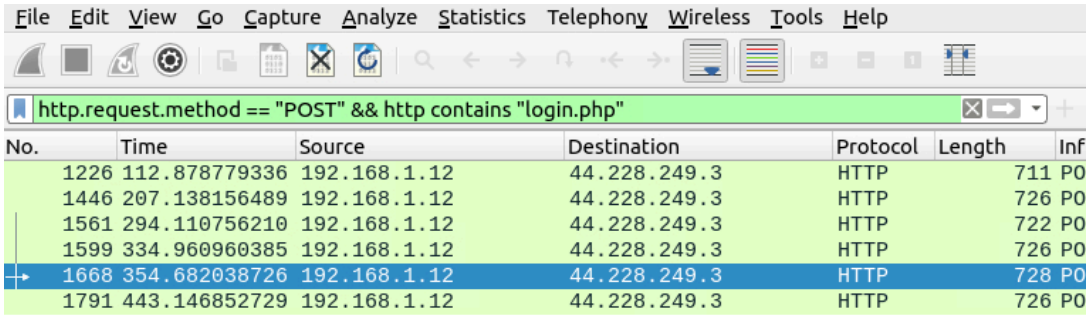


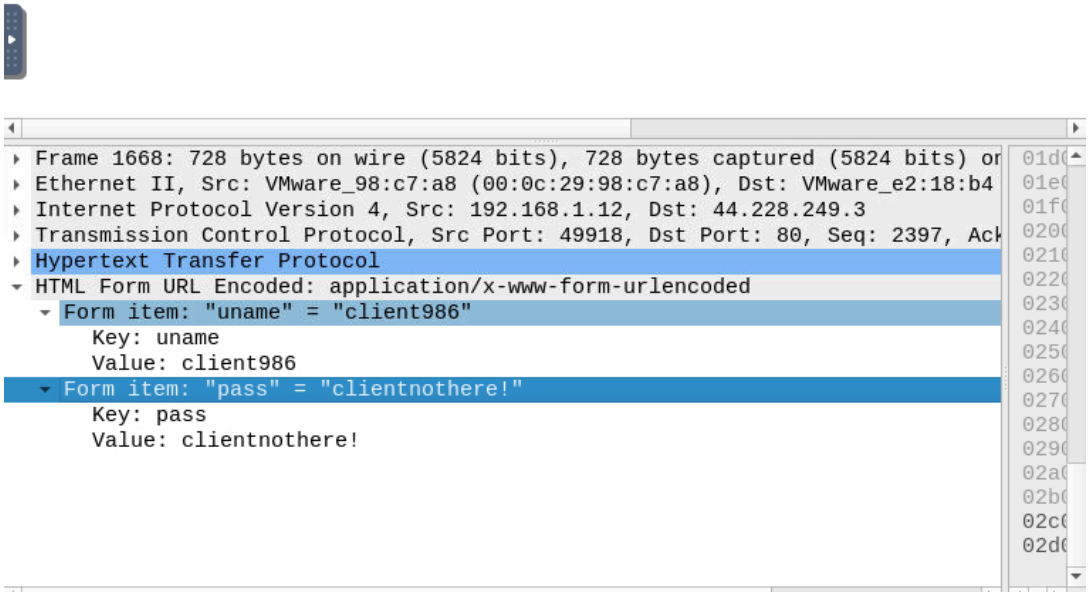
3-What is the number of sniffed username&password entries?

The answer :6

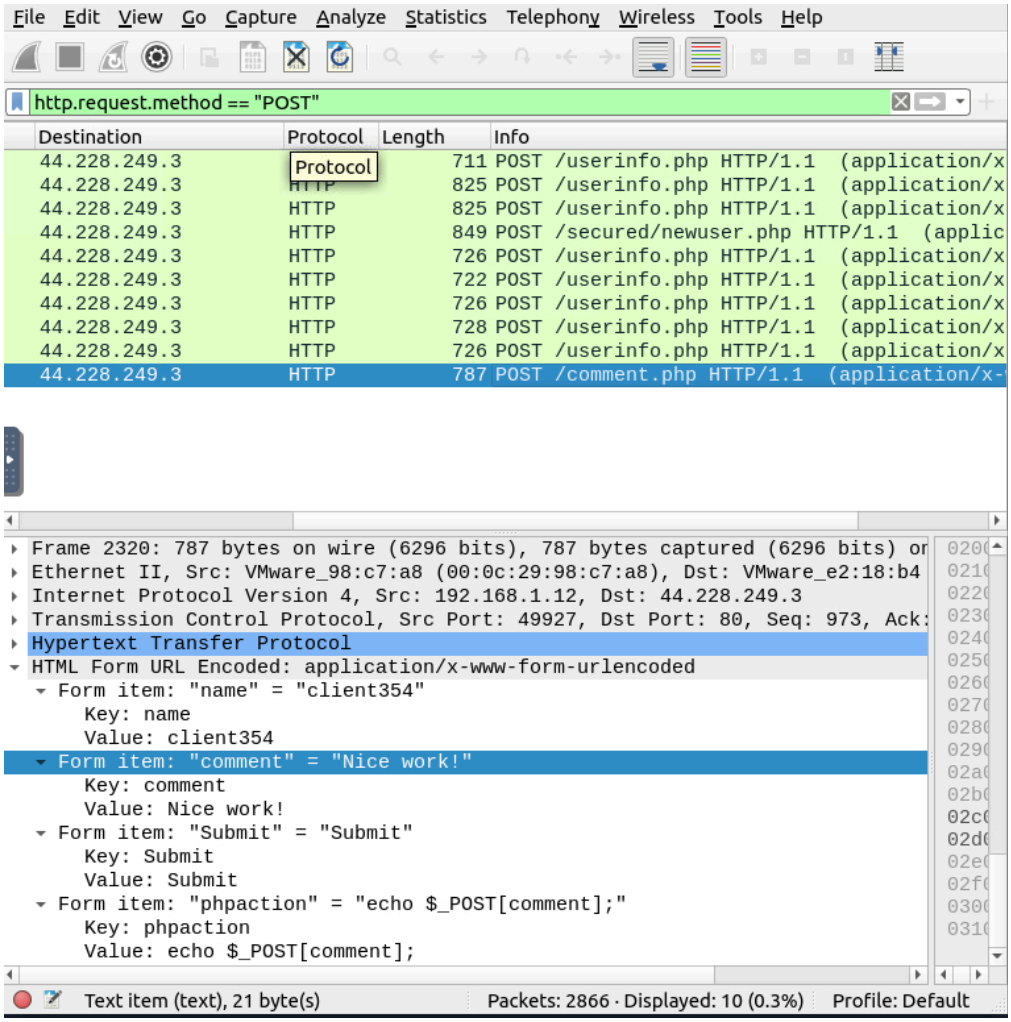


4-What is the password of the "Client986"?





5-What is the comment provided by the "Client354"?



3- Identifying Hosts: DHCP, NetBIOS and Kerberos:

- DHCP

Filtering the proper DHCP packet options is vital to finding an event of interest.

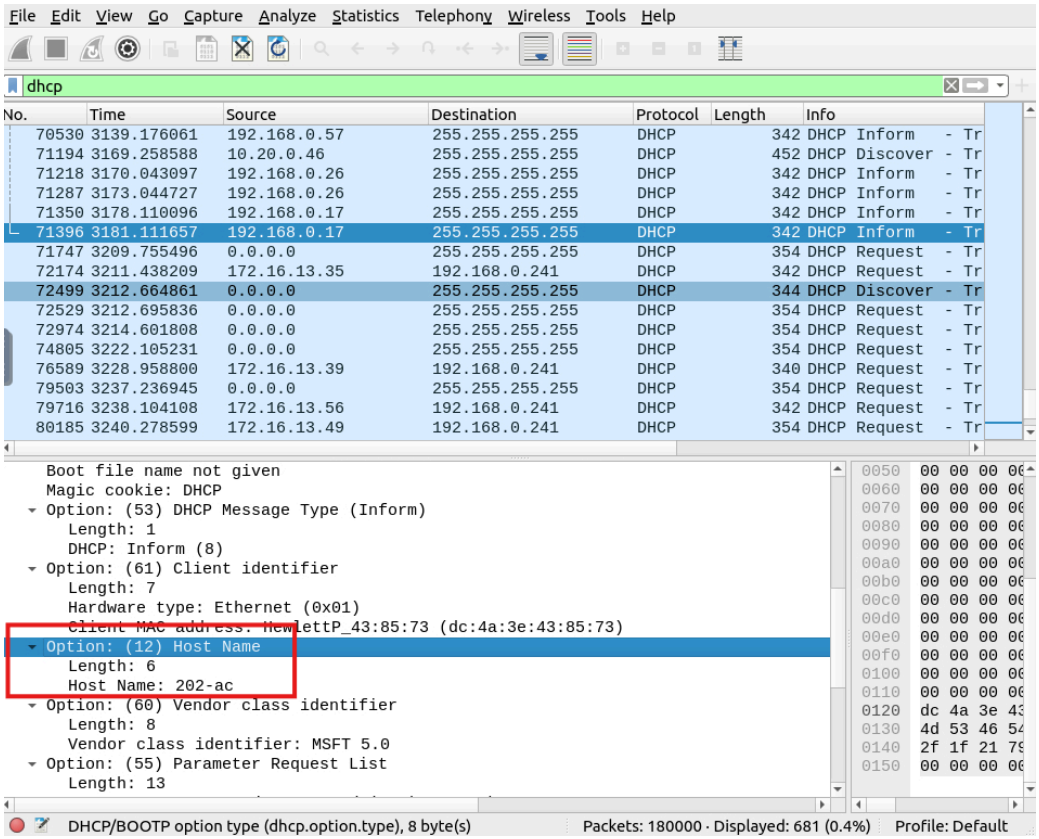
- "DHCP Request" packets contain the hostname information
- "DHCP ACK" packets represent the accepted requests
- "DHCP NAK" packets represent denied requests

Due to the nature of the protocol, only "Option 53" (request type) has predefined static values. You should filter the packet type first, and then you can filter the rest of the options by "applying as column" or use the advanced filters like "contains" and "matches".

- **"DHCP Request"**options for grabbing the low-hanging fruits:
- **Option 12:**Hostname.
- **Option 50:**Requested IP address.
- **Option 51:**Requested IP lease time.
- **Option 61:**Client's MAC address.

- **"DHCP ACK"**options for grabbing the low-hanging fruits:
- **Option 15:**Domain name.
- **Option 51:**Assigned IP lease time.

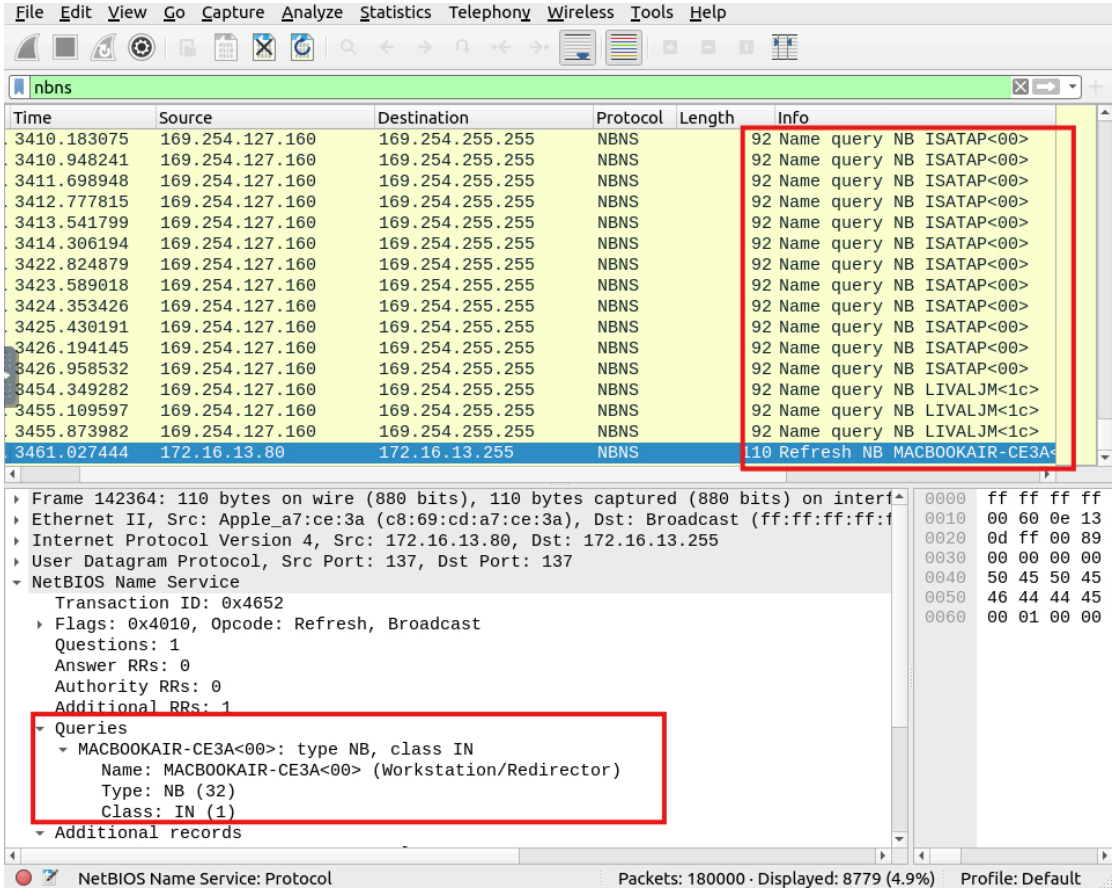
- **"DHCP NAK"** options for grabbing the low-hanging fruits:
- **Option 56:**Message (rejection details/reason).



- NetBIOS :

"NBNS" options for grabbing the low-hanging fruits:

- Queries:Query details.
- Query details could contain "name, Time to live (TTL) and IP address details"



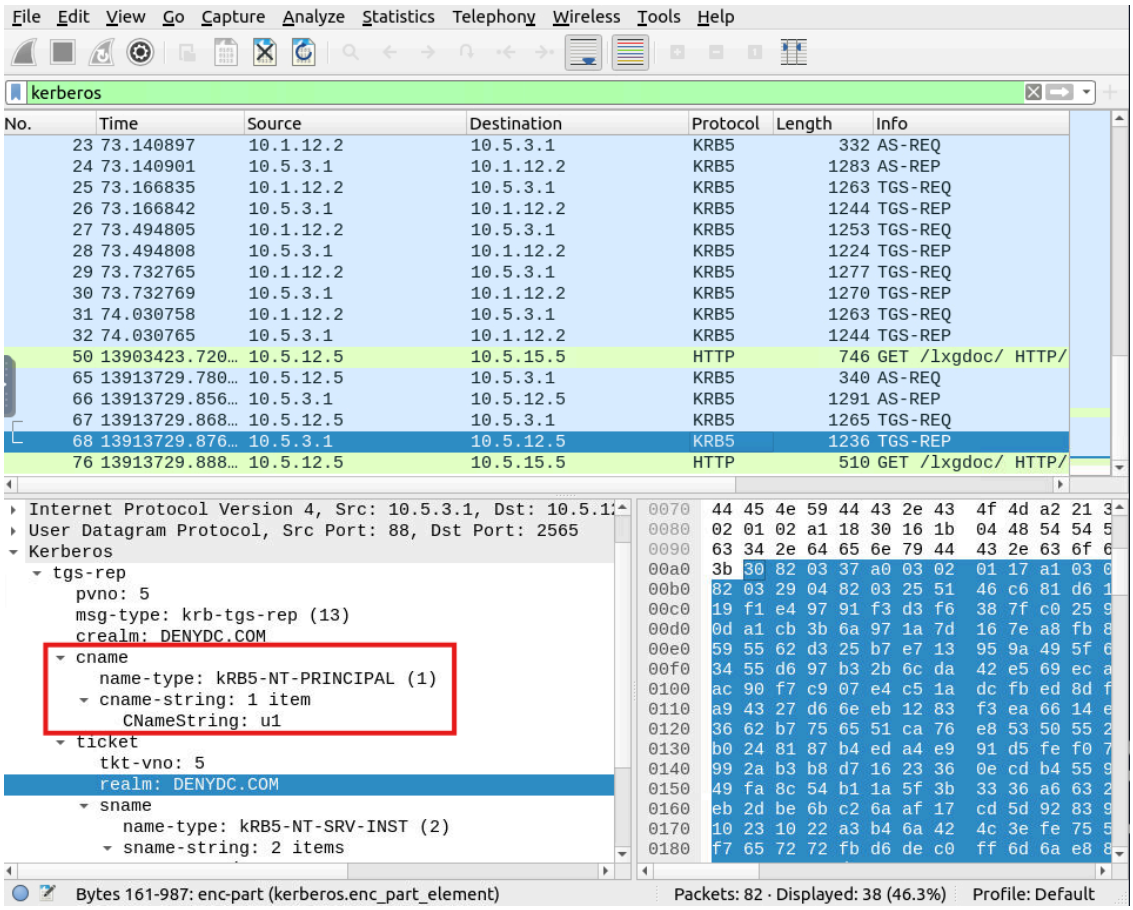
- Kerberos Analysis:

-

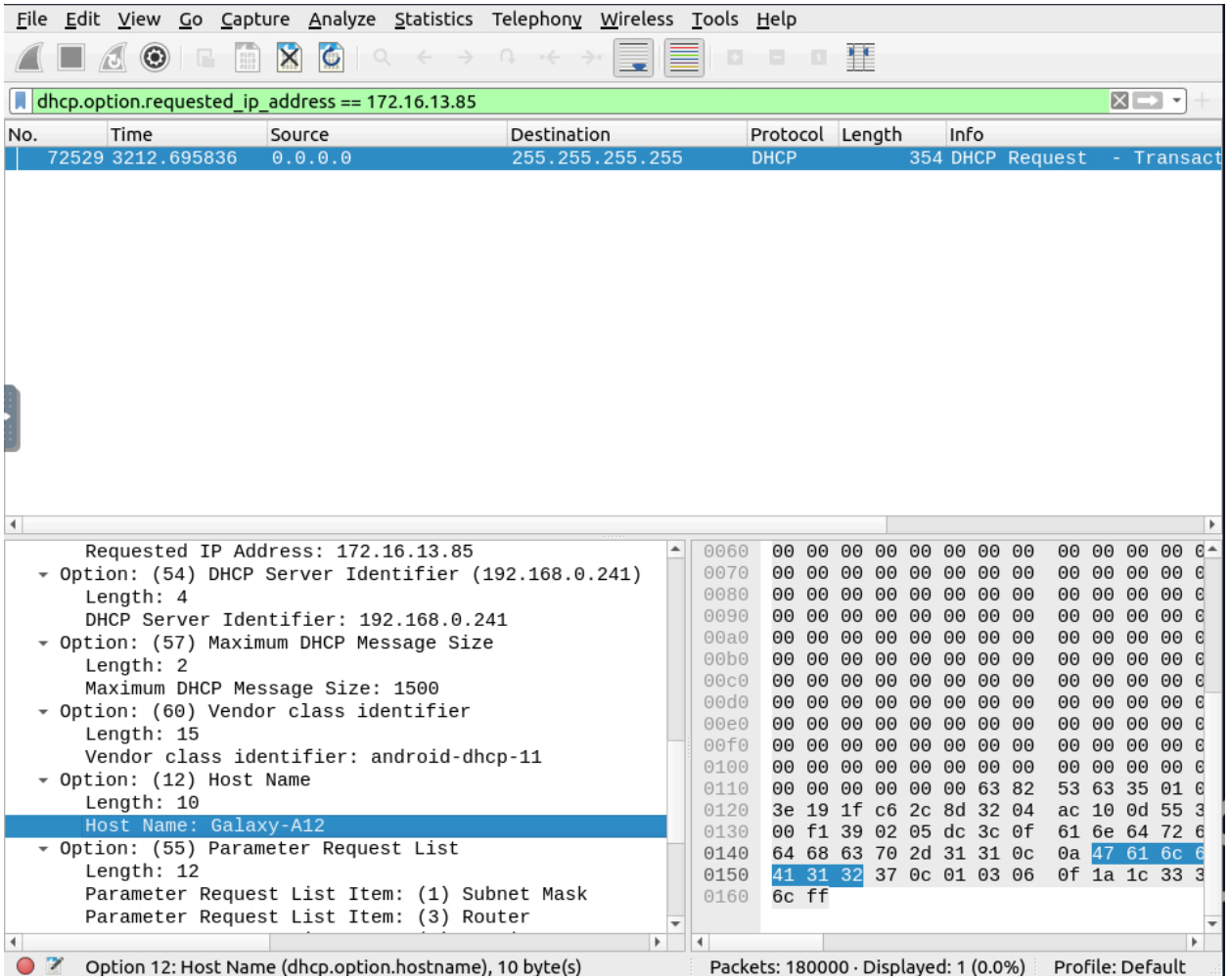
- **Kerberos is the default authentication service for Microsoft Windows domains. It is responsible for authenticating service requests between two or more computers over the untrusted network. The ultimate aim is to prove identity securely.**
- **CNameString:**The username.

Note: Some packets could provide hostname information in this field. To avoid this confusion, filter the **"$"**value. The values end with**"$"**are hostnames, and the ones without it are user names.

- "Kerberos" options for grabbing the low-hanging fruits:
- **pvno:**Protocol version.
- **realm:**Domain name for the generated ticket.
- **sname:**Service and domain name for the generated ticket.
- **addresses:**Client IP address and NetBIOS name.

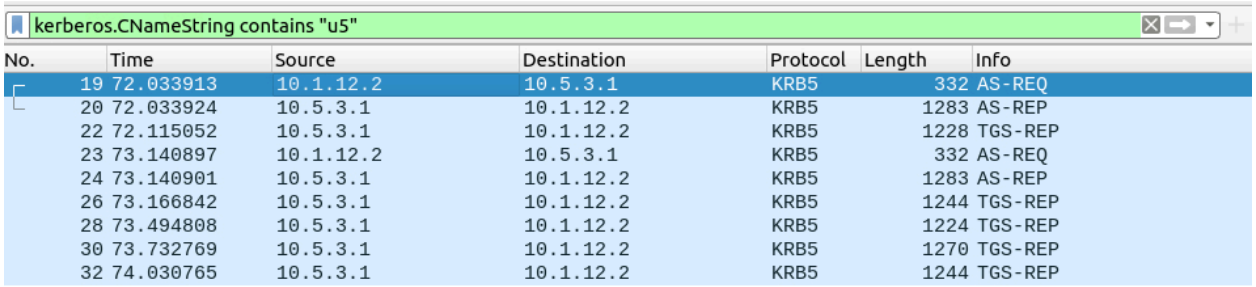


1- Which host requested the IP address "172.16.13.85"?



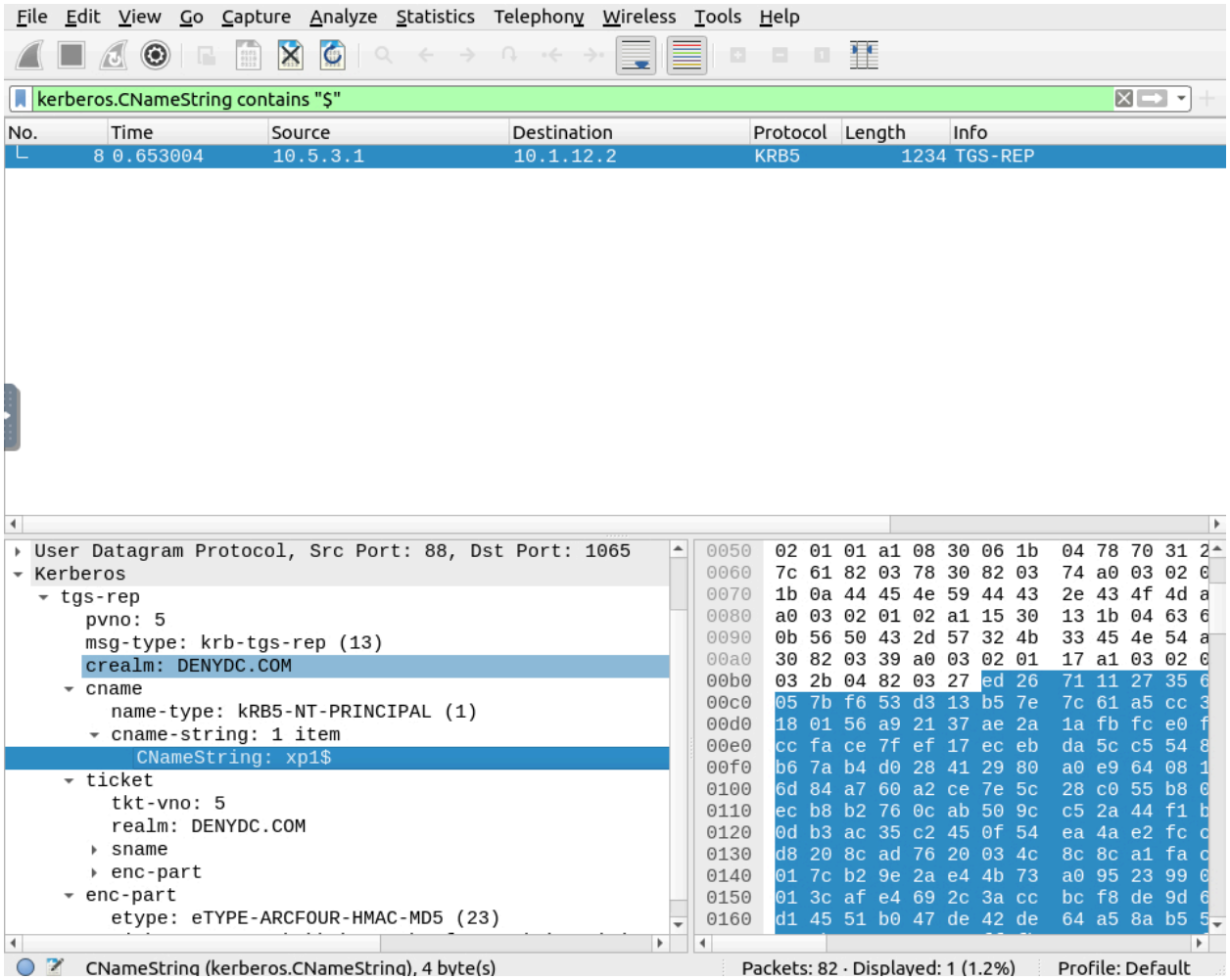
2-What is the IP address of the user "u5"? (Enter the address in defanged format.)

The answer : 10[.]1[.]12[.]2



What is the hostname of the available host in the Kerberos packets?

The answer : xp1$



- Tunnelling Traffic: ICMP and DNS:

Traffic tunnelling is (also known as "port forwarding" transferring the data/resources in a secure method to network segments and zones. It can be used for "internet to private networks" and "private networks to internet" flow/direction. There is an encapsulation process to hide the data, so the transferred data appear natural for the case, but it contains private data packets and transfers them to the final destination securely.

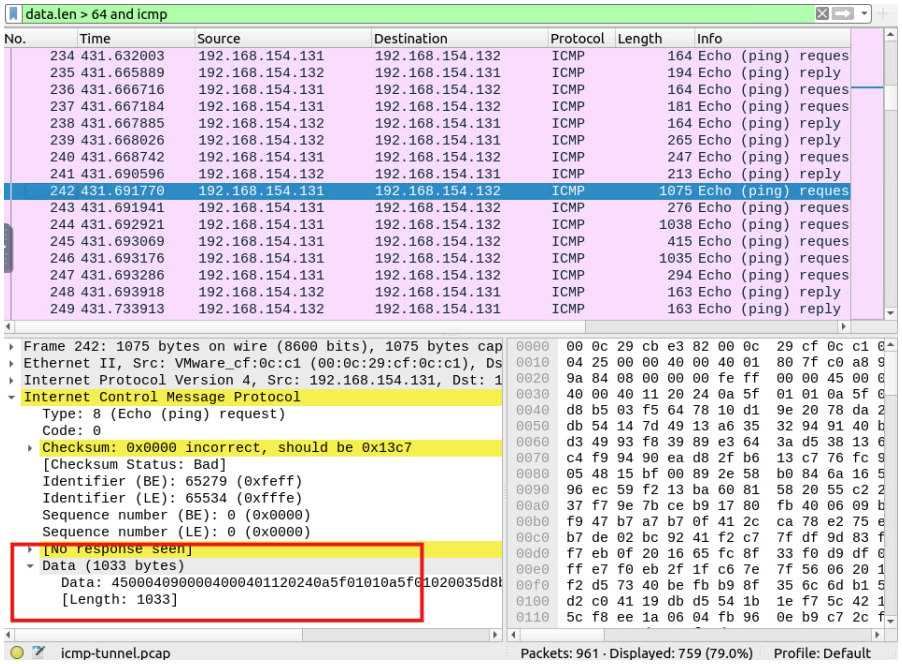
Tunnelling provides anonymity and traffic security. Therefore it is highly used by enterprise networks. However, as it gives a significant level of data encryption, attackers use tunnelling to bypass security perimeters using the standard and trusted protocols used in everyday traffic like ICMP and DNS. Therefore, for a security analyst, it is crucial to have the ability to spot ICMP and DNS anomalies.

ICMP Analysis:

Internet Control Message Protocol (ICMP) is designed for diagnosing and reporting network communication issues. It is highly used in error reporting and testing. As it is a trusted network layer protocol, sometimes it is used for denial of service (DoS) attacks; also, adversaries use it in data exfiltration and C2 tunnelling activities.

"ICMP" options for grabbing the low-hanging fruits:

- Packet length.
- ICMP destination addresses.
- Encapsulated protocol signs in ICMP payload.
- `data.len > 64 and icmp`

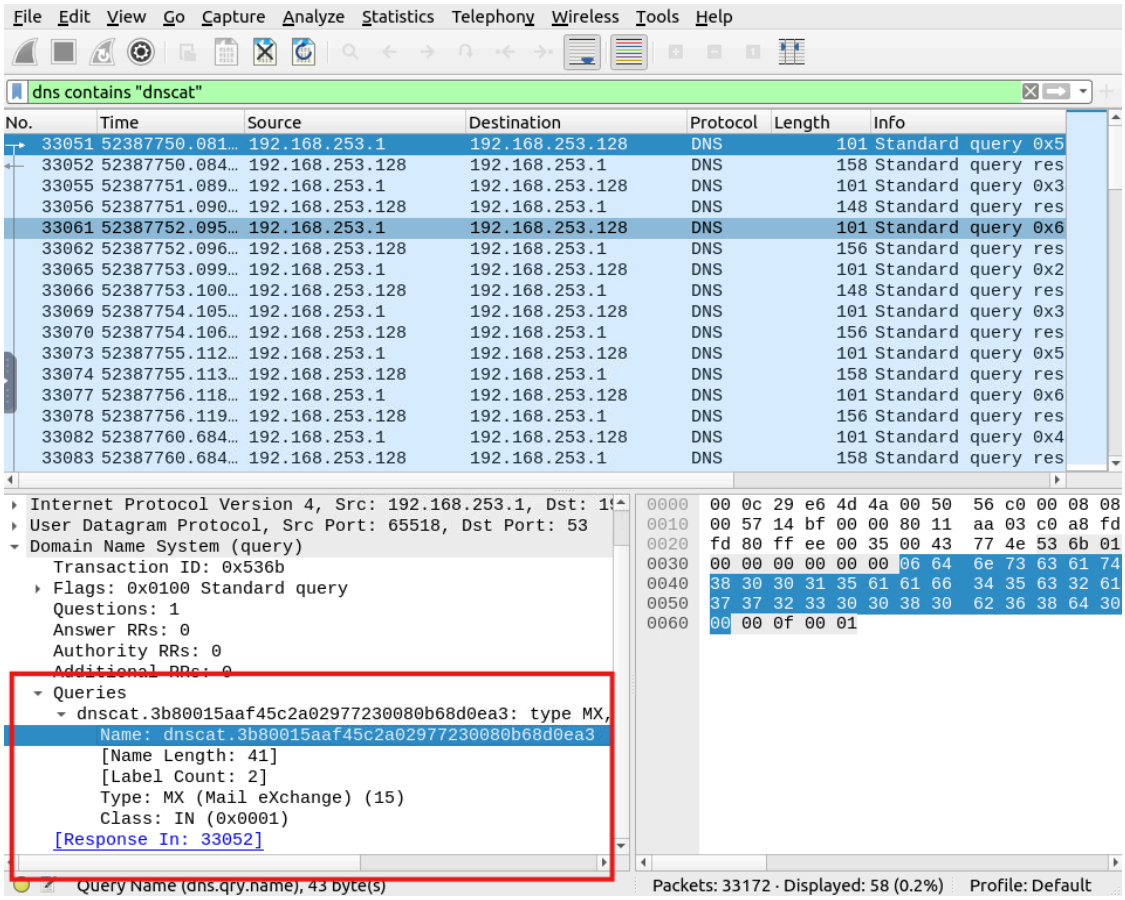


DNS Analysis:

Domain Name System (DNS) is designed to translate/convert IP domain addresses to IP addresses. It is also known as a phonebook of the internet. As it is the essential part of web services, it is commonly used and trusted, and therefore often ignored. Due to that, adversaries use it in data exfiltration and C2 activities.

"DNS" options for grabbing the low-hanging fruits:

- Query length.
- Anomalous and non-regular names in DNS addresses.
- Long DNS addresses with encoded subdomain addresses.
- Known patterns like dnscat and dns2tcp.
- Statistical analysis like the anomalous volume of DNS requests for a particular target.
- `dns contains "dnscat"`
- `dns.qry.name.len > 15 and !mdns`



- **Cleartext Protocol Analysis: FTP**

  File Transfer Protocol (FTP) is designed to transfer files with ease, so it focuses on simplicity rather than security. As a result of this, using this protocol in unsecured environments could create security issues like:

- MITM attacks
- Credential stealing and unauthorised access
- Phishing
- Malware planting
- Data exfiltration

- "FTP" options for grabbing the low-hanging fruits:
- x1x series: Information request responses.
- x2x series: Connection messages.
- x3x series: Authentication messages.

Note: "200" means command successful.

- **"x1x" series options for grabbing the low-hanging fruits:**
- **211: System status.**
- **212: Directory status.**
- **213: File status**
- `ftp.response.code == 211`

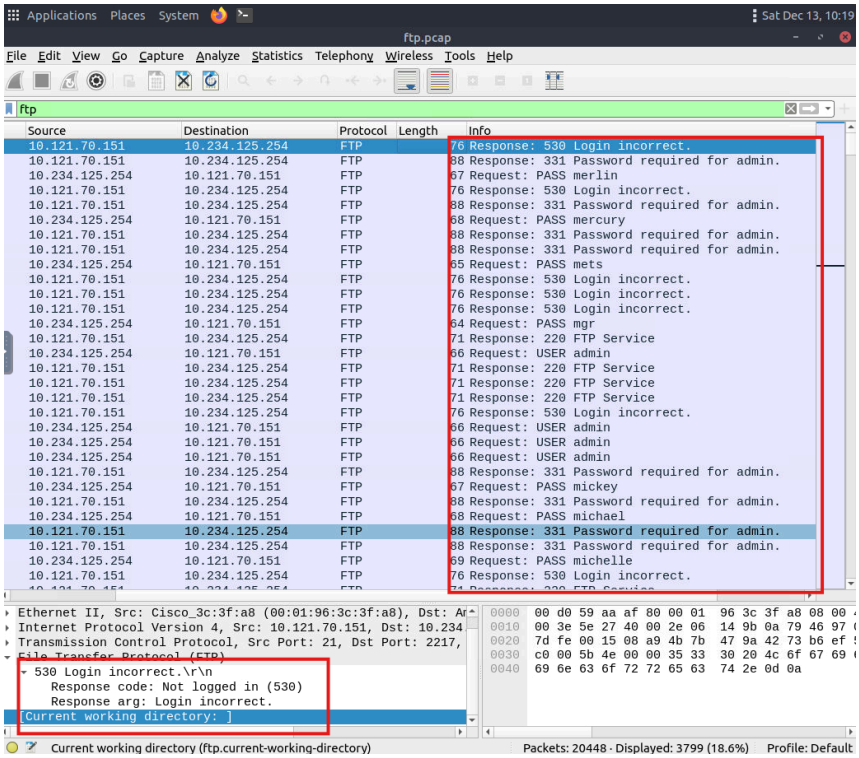
- "x2x" series options for grabbing the low-hanging fruits:
- 220: Service ready.
- 227: Entering passive mode.
- 228: Long passive mode.
- 229: Extended passive mode.
- `ftp.response.code == 227`

- "x3x" series options for grabbing the low-hanging fruits:
- 230: User login.
- 231: User logout.
- 331: Valid username.
- 430: Invalid username or password
- 530: No login, invalid password.
- `ftp.response.code == 230`

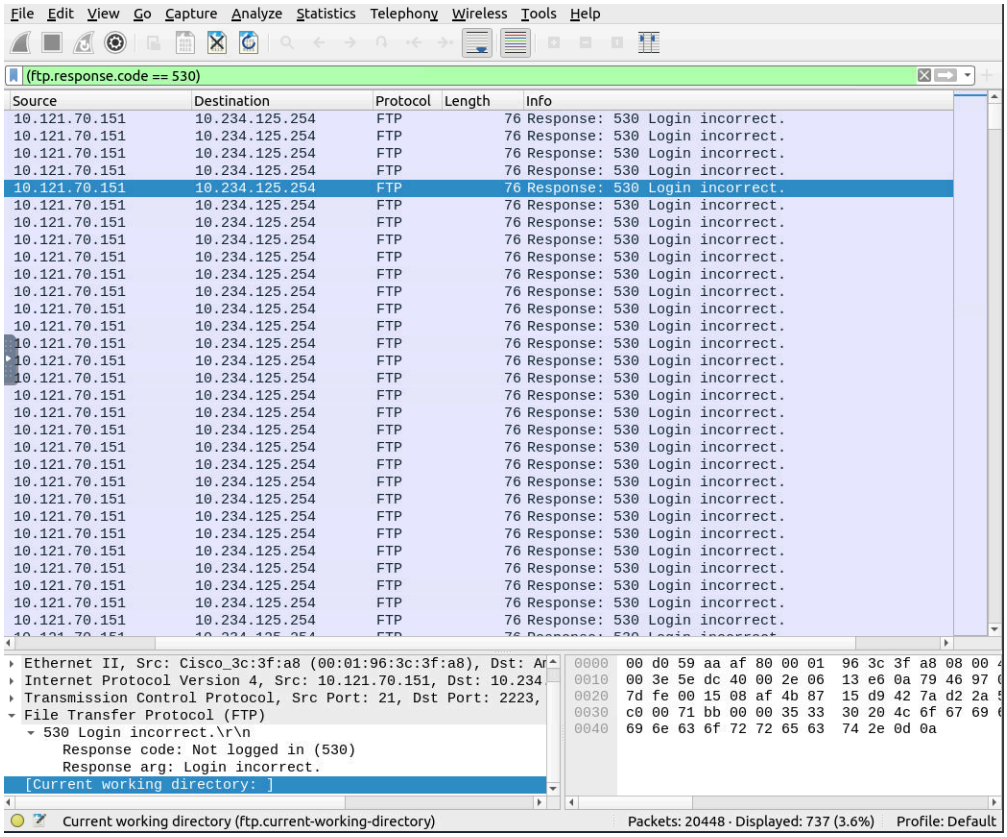
- "FTP" commands for grabbing the low-hanging fruits:
- USER: Username.
- PASS: Password.
- CWD: Current work directory.
- LIST: List.

- `ftp.request.command == "USER"`
- `ftp.request.command == "PASS"`
- `ftp.request.arg == "password"`

- Advanced usages examples for grabbing low-hanging fruits:
- Bruteforce signal: List failed login attempts.
- Bruteforce signal: List target username.
- Password spray signal: List targets for a static password.
- `ftp.response.code == 530`
- `(ftp.response.code == 530) and (ftp.response.arg contains "username")`
- `(ftp.request.command == "PASS" ) and (ftp.request.arg == "password")`


Applications Places System
Sat Dec 13, 10:19
ftp.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ftp
Source Destination Protocol Length Info
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.234.125.254 10.121.70.151 FTP 67 Request: PASS merlin
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.234.125.254 10.121.70.151 FTP 68 Request: PASS mercury
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.234.125.254 10.121.70.151 FTP 65 Request: PASS mets
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
10.234.125.254 10.121.70.151 FTP 64 Request: PASS mgr
10.121.70.151 10.234.125.254 FTP 71 Response: 220 FTP Service
10.234.125.254 10.121.70.151 FTP 66 Request: USER admin
10.121.70.151 10.234.125.254 FTP 71 Response: 220 FTP Service
10.121.70.151 10.234.125.254 FTP 71 Response: 220 FTP Service
10.121.70.151 10.234.125.254 FTP 71 Response: 220 FTP Service
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
10.234.125.254 10.121.70.151 FTP 66 Request: USER admin
10.234.125.254 10.121.70.151 FTP 66 Request: USER admin
10.234.125.254 10.121.70.151 FTP 66 Request: USER admin
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.234.125.254 10.121.70.151 FTP 67 Request: PASS mickey
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.234.125.254 10.121.70.151 FTP 68 Request: PASS michael
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.121.70.151 10.234.125.254 FTP 88 Response: 331 Password required for admin.
10.234.125.254 10.121.70.151 FTP 69 Request: PASS michelle
10.121.70.151 10.234.125.254 FTP 76 Response: 530 Login incorrect.
Ethernet II, Src: Cisco_3c:3f:a8 (00:01:96:3c:3f:a8), Dst: Ar
Internet Protocol Version 4, Src: 10.121.70.151, Dst: 10.234
Transmission Control Protocol, Src Port: 21, Dst Port: 2217,
File Transfer Protocol (FTP)
530 Login incorrect.\r\n
Response code: Not logged in (530)
Response arg: Login incorrect.
[Current working directory:]
0000 00 d0 59 aa af 80 00 01 96 3c 3f a8 08 00
0010 00 3e 5e 27 40 00 2e 06 14 9b 0a 79 46 97
0020 7d fe 00 15 08 a9 4b 7b 47 9a 42 73 b6 ef
0030 c0 00 5b 4e 00 00 35 33 30 20 4c 6f 67 69
0040 69 6e 63 6f 72 72 65 63 74 2e 0d 0a
Current working directory (ftp.current-working-directory)
Packets: 20448 · Displayed: 3799 (18.6%)
Profile: Default

- **Bruteforce signal: List failed login attempts.**



- Cleartext Protocol Analysis: HTTP

Hypertext Transfer Protocol (HTTP) is a cleartext-based, request-response and client-server protocol. It is the standard type of network activity to request/serve web pages, and by default, it is not blocked by any network perimeter. As a result of being unencrypted and the backbone of web traffic, HTTP is one of the must-to-know protocols in traffic analysis. Following attacks could be detected with the help of HTTP analysis:

- Phishing pages
- Web attacks
- Data exfiltration
- Command and control traffic (C2)

- "HTTP Request Methods" for grabbing the low-hanging fruits:
- GET
- POST
- Request: Listing all requests

- "HTTP Response Status Codes" for grabbing the low-hanging fruits:
- 200 OK: Request successful.
- 301 Moved Permanently: Resource is moved to a new URL/path (permanently).
- 302 Moved Temporarily: Resource is moved to a new URL/path (temporarily).
- 400 Bad Request: Server didn't understand the request.
- 401 Unauthorised: URL needs authorisation (login, etc.).
- 403 Forbidden: No access to the requested URL.
- 404 Not Found: Server can't find the requested URL.
- 405 Method Not Allowed: Used method is not suitable or blocked.
- 408 Request Timeout: Request look longer than server wait time.
- 500 Internal Server Error: Request not completed, unexpected error.
- 503 Service Unavailable: Request not completed server or service is down.

- "HTTP Parameters" for grabbing the low-hanging fruits:
- User agent: Browser and operating system identification to a web server application.

- Request URI: Points the requested resource from the server.
- Full *URI: Complete URI information.

*URI: Uniform Resource Identifier.

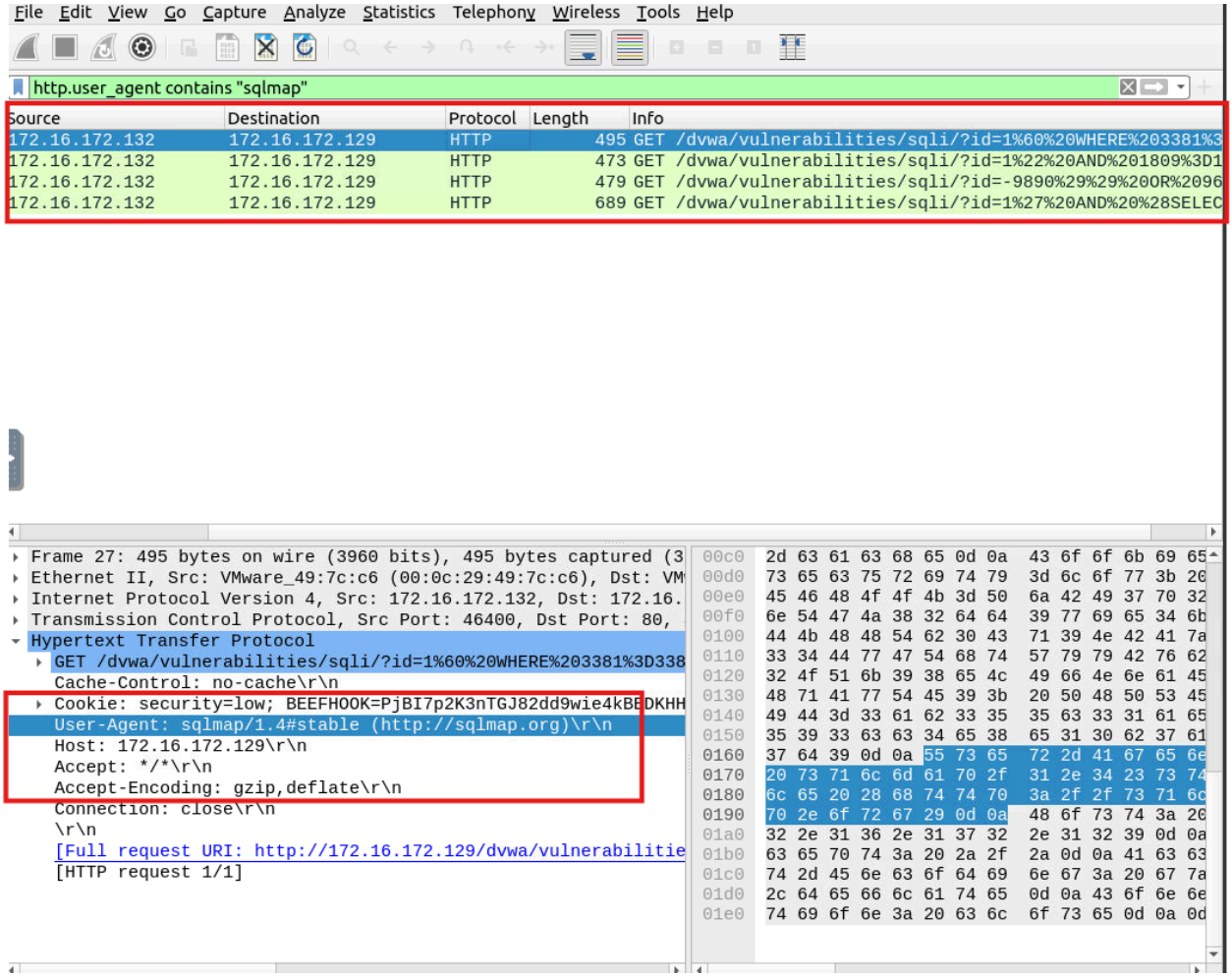
User Agent Analysis:

As the adversaries use sophisticated technics to accomplish attacks, they try to leave traces similar to natural traffic through the known and trusted protocols. For a security analyst, it is important to spot the anomaly signs on the bits and pieces of the packets. The "user-agent" field is one of the great resources for spotting anomalies in HTTP traffic. In some cases, adversaries successfully modify the user-agent data, which could look super natural. A security analyst cannot rely only on the user-agent field to spot an anomaly. Never whitelist a user agent, even if it looks natural. User agent-based anomaly/threat detection/hunting is an additional data source to check and is useful when there is an obvious anomaly.

Research outcomes for grabbing the low-hanging fruits:

- Different user agent information from the same host in a short time notice.
- Non-standard and custom user agent info.
- Subtle spelling differences. ("Mozilla" is not the same as "Mozlilla" or "Mozlila")
- Audit tools info like Nmap, Nikto, Wfuzz and sqlmap in the user agent field.
- Payload data in the user agent field.

- `(http.user_agent contains "sqlmap") or (http.user_agent contains "Nmap") or (http.user_agent contains "Wfuzz") or (http.user_agent contains "Nikto")`


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http.user_agent contains "sqlmap"
Source Destination Protocol Length Info
172.16.172.132 172.16.172.129 HTTP 495 GET /dvwa/vulnerabilities/sqli/?id=1%60%20WHERE%203381%3
172.16.172.132 172.16.172.129 HTTP 473 GET /dvwa/vulnerabilities/sqli/?id=1%22%20AND%201809%3D1
172.16.172.132 172.16.172.129 HTTP 479 GET /dvwa/vulnerabilities/sqli/?id=-9890%29%29%20OR%2096
172.16.172.132 172.16.172.129 HTTP 689 GET /dvwa/vulnerabilities/sqli/?id=1%27%20AND%20%28SELEC
Frame 27: 495 bytes on wire (3960 bits), 495 bytes captured (3
Ethernet II, Src: VMware_49:7c:c6 (00:0c:29:49:7c:c6), Dst: VM
Internet Protocol Version 4, Src: 172.16.172.132, Dst: 172.16.
Transmission Control Protocol, Src Port: 46400, Dst Port: 80,
Hypertext Transfer Protocol
GET /dvwa/vulnerabilities/sqli/?id=1%60%20WHERE%203381%3D338
Cache-Control: no-cache\r\n
Cookie: security=low; BEEFHOOK=PjBI7p2K3nTGJ82dd9wie4kBEDKHH
User-Agent: sqlmap/1.4#stable (http://sqlmap.org)\r\n
Host: 172.16.172.129\r\n
Accept: */*\r\n
Accept-Encoding: gzip,deflate\r\n
Connection: close\r\n
\r\n
[Full request URI: http://172.16.172.129/dvwa/vulnerabilitie
[HTTP request 1/1]
00c0 2d 63 61 63 68 65 0d 0a 43 6f 6f 6b 69 65
00d0 73 65 63 75 72 69 74 79 3d 6c 6f 77 3b 20
00e0 45 46 48 4f 4f 4b 3d 50 6a 42 49 37 70 32
00f0 6e 54 47 4a 38 32 64 64 39 77 69 65 34 6b
0100 44 4b 48 48 54 62 30 43 71 39 4e 42 41 7a
0110 33 34 44 77 47 54 68 74 57 79 79 42 76 62
0120 32 4f 51 6b 39 38 65 4c 49 66 4e 6e 6e 61 45
0130 48 71 41 77 54 45 39 3b 20 50 48 50 53 45
0140 49 44 3d 33 61 62 33 35 35 63 33 31 61 65
0150 35 39 33 63 63 34 65 38 65 31 30 62 37 61
0160 37 64 39 0d 0a 55 73 65 72 2d 41 67 65 6e
0170 20 73 71 6c 6d 61 70 2f 31 2e 34 23 73 74
0180 6c 65 20 28 68 74 74 70 3a 2f 2f 73 71 6c
0190 70 2e 6f 72 67 29 0d 0a 48 6f 73 74 3a 20
01a0 32 2e 31 36 2e 31 37 32 2e 31 32 39 0d 0a
01b0 63 65 70 74 3a 20 2a 2f 2a 0d 0a 41 63 63
01c0 74 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a
01d0 2c 64 65 66 6c 61 74 65 0d 0a 43 6f 6e 6e
01e0 74 69 6f 6e 3a 20 63 6c 6f 73 65 0d 0a 0d