

Firewall Evasion

Task 0: Lab Setup

1. Set up hosts:

```
[11/14/24]seed@VM:~/.../Labsetup$ dcup
Creating network "net-10.8.0.0" with the default driver
Creating network "net-192.168.20.0" with the default driver
Building hostA
Step 1/5 : FROM handsonsecurity/seed-ubuntu:large
large: Pulling from handsonsecurity/seed-ubuntu
da7391352a9b: Pulling fs layer
14428a6d4bcd: Downloading [=====>]
14428a6d4bcd: Downloading [=====>]
da7391352a9b: Downloading [>]
da7391352a9b: Downloading [=====>]
2.345MB/28.56MBwnload complete
da7391352a9b: Pull complete
14428a6d4bcd: Pull complete
2c2d948710f2: Pull complete
b5e99359ad22: Pull complete
Attaching to B2-192.168.20.6, A1-10.8.0.5, A-10.8.0.99, A2-10.8.0.6, B-192.168.2
0.99, B1-192.168.20.5, router-firewall
A-10.8.0.99 | * Starting internet superserver inetd [ OK ]
A1-10.8.0.5 | * Starting internet superserver inetd [ OK ]
B-192.168.20.99 | * Starting internet superserver inetd [ OK ]
A2-10.8.0.6 | * Starting internet superserver inetd [ OK ]
B2-192.168.20.6 | * Starting internet superserver inetd [ OK ]
B1-192.168.20.5 | * Starting internet superserver inetd [ OK ]
A-10.8.0.99 | * Starting OpenBSD Secure Shell server sshd [ OK ]
router-firewall | * Starting internet superserver inetd [ OK ]
B-192.168.20.99 | * Starting OpenBSD Secure Shell server sshd [ OK ]
```

2. Display hosts:

```
[11/14/24]seed@VM:~/.../Labsetup$ dockps
ba5a4a4ce992 A1-10.8.0.5
d07470d6d9e2 B-192.168.20.99
12601cd144d5 router-firewall
2ae70a4ebde6 A-10.8.0.99
4cd208a3ce0f B1-192.168.20.5
0d800bcc2b91 B2-192.168.20.6
412021870aa2 A2-10.8.0.6
```

Task 1: Static Port Forwarding

1. Set up ssh command for port connection (external port A to internal port B1):

```
[11/14/24]seed@VM:~/.../Labsetup$ docksh 2ae
root@2ae70a4ebde6:/# ssh -4NT -L 10.8.0.99:7000:192.168.20.5:23 seed@192.168.20.99
The authenticity of host '192.168.20.99 (192.168.20.99)' can't be established.
ECDSA key fingerprint is SHA256:528d4rVKUV+Z5V1J1kVzuKmIZ8mjcyCYAtwcbtdpvYA.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.20.99' (ECDSA) to the list of known hosts.
seed@192.168.20.99's password:
```

2. Connecting to port B1 through root port A1:

```
[11/14/24]seed@VM:~/.../Labsetup$ docksh ba5
root@ba5a4a4ce992:/# telnet 10.8.0.99 7000
Trying 10.8.0.99...
Connected to 10.8.0.99.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4cd208a3ce0f login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@4cd208a3ce0f:~$
```

3. Connecting to port B1 through root port A2:

```
[11/14/24]seed@VM:~/../Labsetup$ docksh 412
root@412021870aa2:/# telnet 10.8.0.99 7000
Trying 10.8.0.99...
Connected to 10.8.0.99.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4cd208a3ce0f login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Nov 14 18:17:01 UTC 2024 from B-192.168.20.99.net-192.168.20.0 on pts/1
seed@4cd208a3ce0f:~$ exit
logout
Connection closed by foreign host.
root@412021870aa2:/# █
```

4. Question About Task 1

Question 1: How many TCP connections are involved in this entire process?

3 connections: Host A (outside the firewall) and Host B (inside the firewall). This connection is like a secure tunnel created by SSH, that allows us to send data into the protected area (the internal network). Once the secure tunnel is set up, any data we send into it from Host A is automatically forwarded through to Host B's internal network. Once the data reaches Host B, it continues to reach a service running on another computer in the internal network (Host B1).

Question 2: Why can this tunnel successfully help users evade the firewall rule specified in the lab setup?

Answer: The setup for this lab firewall allows SSH traffic to go through. Running the SSH command allows our data to show up as SSH traffic, which the firewall already allows. By creating a secure tunnel (using SSH) from Host A (outside) to Host B (inside), we send the data through the firewall disguised as SSH. Since the firewall doesn't know what's inside this SSH tunnel, it lets the data pass, thinking it's just a regular allowed SSH connection.

Task 2: Dynamic Port Forwarding

1. Task 2.1: Setting Up Dynamic Port Forwarding (connect to host A through host B)

```
[11/14/24]seed@VM:~/.../Labsetup$ dockps
ba5a4a4ce992    A1-10.8.0.5
d07470d6d9e2    B-192.168.20.99
12601cd144d5    router-firewall
2ae70a4ebde6    A-10.8.0.99
4cd208a3ce0f    B1-192.168.20.5
0d800bcc2b91    B2-192.168.20.6
412021870aa2    A2-10.8.0.6
[11/14/24]seed@VM:~/.../Labsetup$ docksh d07
root@d07470d6d9e2:/# ssh -4NT -D 192.168.20.99:7200 seed@10.8.0.99
The authenticity of host '10.8.0.99 (10.8.0.99)' can't be established.
ECDSA key fingerprint is SHA256:528d4rVKUV+Z5V1J1kVzuKmIZ8mjcyCYAtwcbtdpvYA.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.8.0.99' (ECDSA) to the list of known hosts.
seed@10.8.0.99's password:
```

2. Connect to example.com using curl command through port B1:

```
root@4cd208a3ce0f:~# curl --proxy socks5h://192.168.20.99:7200 www.example.com
<!doctype html>
<html>
<head>
  <title>Example Domain</title>

  <meta charset="utf-8" />
  <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <body type="text/css">
    body {
      background-color: #f0f0f2;
      margin: 0;
      padding: 0;
      font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;
    }
    div {
      width: 600px;
      margin: 5em auto;
      padding: 2em;
      background-color: #fdfdff;
      border-radius: 0.5em;
      box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
      color: #38488f;
      text-decoration: none;
    }
    @media (max-width: 700px) {
      div {

```

3. Connect to example.com using curl command through port B2:

```
[11/14/24]seed@VM:~/.../Labsetup$ docksh 0d8
root@0d800bcc2b91:/# curl --proxy socks5h://192.168.20.99:7200 www.example.com
<!doctype html>
<html>
<head>
  <title>Example Domain</title>

  <meta charset="utf-8" />
  <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <style type="text/css">
    body {
      background-color: #f0f0f2;
      margin: 0;
      padding: 0;
      font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;
    }
    div {
      width: 600px;
      margin: 5em auto;
      padding: 2em;
      background-color: #fdfdff;
      border-radius: 0.5em;
      box-shadow: 2px 3px 7px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
      color: #38488f;
      text-decoration: none;
    }
  </style>
</head>
<body>
  <div>
    <h1>Example Domain</h1>
    <p>This domain is for use in illustrative examples in documents. You may use this domain in literature without prior coordination or asking for permission.</p>
    <pre><code></code></pre>
  </div>
</body>
</html>
```

4. Questions About Task 2.1

Question 1: Which computer establishes the actual connection with the intended web server?

Answer: In this setup, **Host A** (outside the firewall) is the computer that establishes the actual connection with the blocked websites. Even though you're accessing the sites from inside the firewall (using Host B or other internal computers), the data goes through the dynamic port forwarding tunnel to Host A. Host A then connects to the websites directly, effectively "hiding" the real origin of the request from the firewall.

Question 2: How does this computer know which server it should connect to?

Answer: Host A knows which website to connect to because **Host B (the proxy)** tells it where to go. We send a request to a blocked site from Host B, it uses the dynamic port forwarding tunnel, which includes information about the website we want to visit. Host B sends the destination through the tunnel to Host A. Host A reads this information and connects to the blocked website directly on our behalf. So, Host B passes data through and sends the address of the website we are visiting, and Host A uses that to know where to connect.

Task 2.2: Testing the Tunnel Using Browser

1. Dropping the IP Address for github.com after connecting to firewall router using docksh f9c:

```
root@f9c53d8316e8:/# iptables -A FORWARD -d 140.82.113.3 -j DROP
root@f9c53d8316e8:/# iptables -L FORWARD
Chain FORWARD (policy ACCEPT)
target     prot opt source                destination              ctstate RELATED,ESTABLISHED
ACCEPT    tcp  --  anywhere              anywhere                  tcp dpt:ssh
DROP      tcp  --  anywhere              anywhere
DROP      all  --  anywhere              93.184.216.0/24
DROP      all  --  anywhere              lb-140-82-113-3-iad.github.com
root@f9c53d8316e8:/# iptables -L FORWARD -n
Chain FORWARD (policy ACCEPT)
target     prot opt source                destination              ctstate RELATED,ESTABLISHED
ACCEPT    tcp  --  0.0.0.0/0             0.0.0.0/0                tcp dpt:22
DROP      tcp  --  0.0.0.0/0             0.0.0.0/0
DROP      all  --  0.0.0.0/0             93.184.216.0/24
DROP      all  --  0.0.0.0/0             140.82.113.3
```

2. Dynamic port forwarding (port 7200) from proxy B to A:

```
root@aa5b1a15ad36:/# ssh -4NT -D 192.168.20.99:7200 seed@10.8.0.99
```

3. Change proxy connection to port B:

Configure Proxy Access to the Internet

☐ No proxy

☐ Auto-detect proxy settings for this network

☐ Use system proxy settings

☒ Manual proxy configuration

HTTP Proxy Port

☐ Also use this proxy for FTP and HTTPS

HTTPS Proxy Port

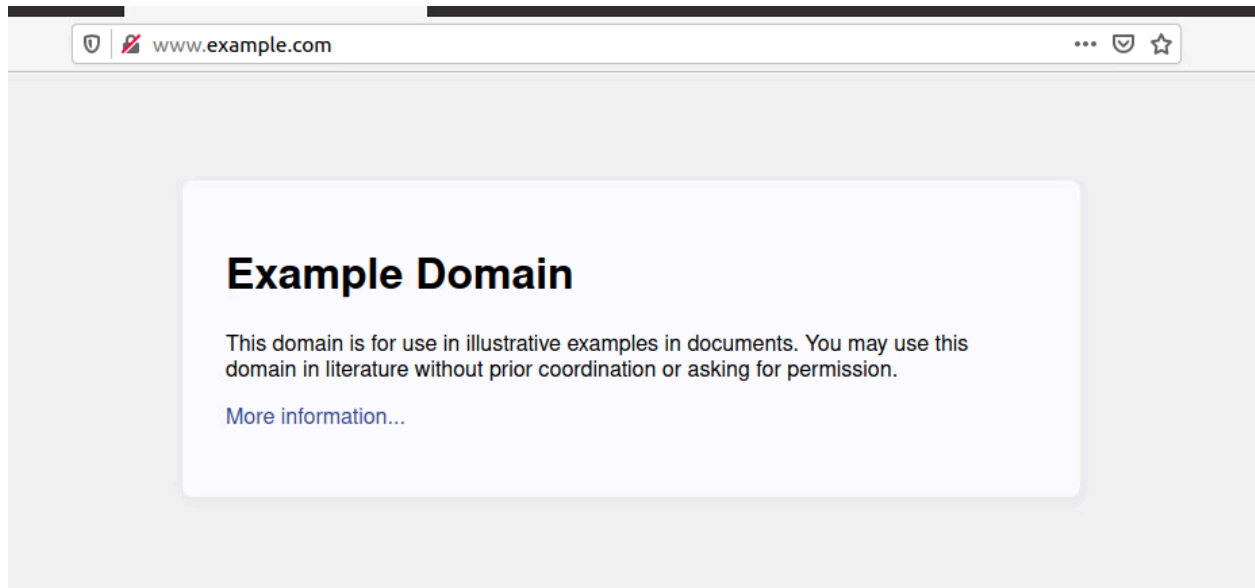
FTP Proxy Port

SOCKS Host Port

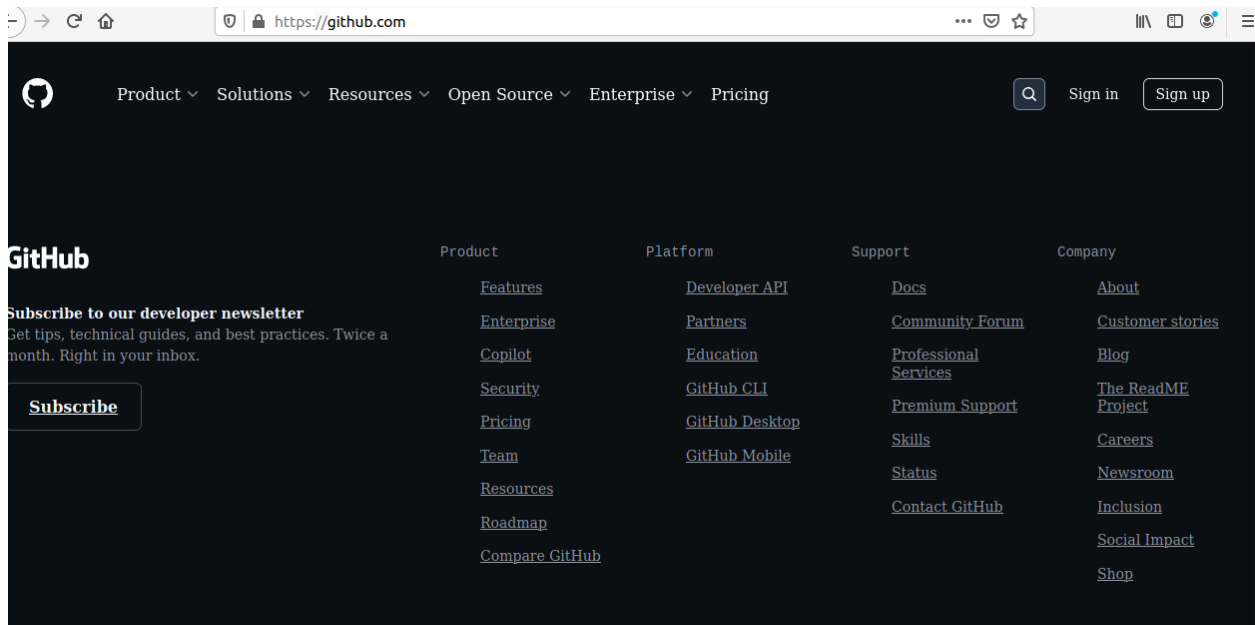
☐ SOCKS v4 ☒ SOCKS v5

☐ Automatic proxy configuration URL

4. Check Example domain - successfully connected through the proxy



5. Check chosen website (github.com) - successfully connected through the proxy

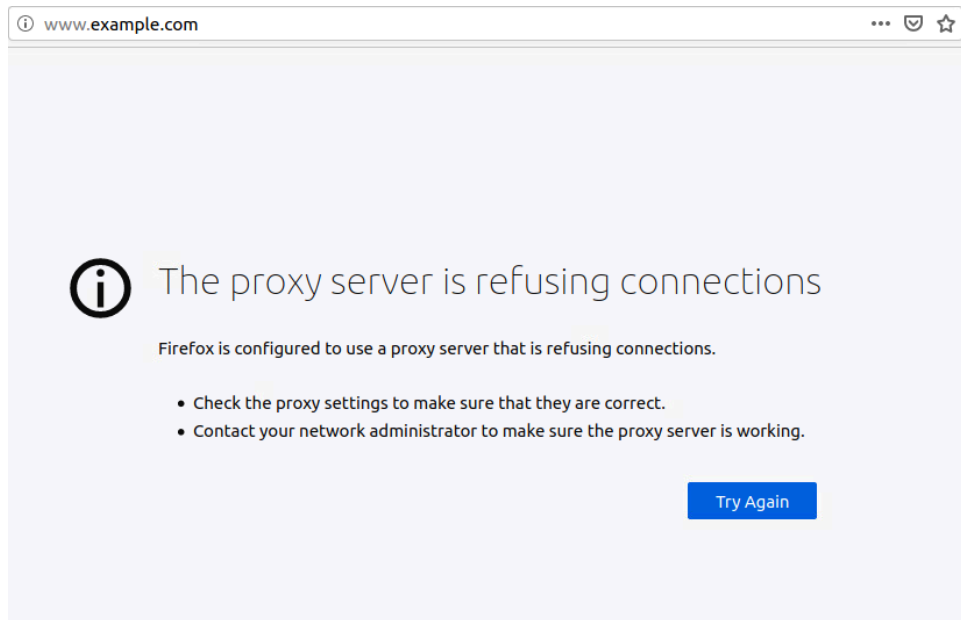


6. Disconnect from the proxy:

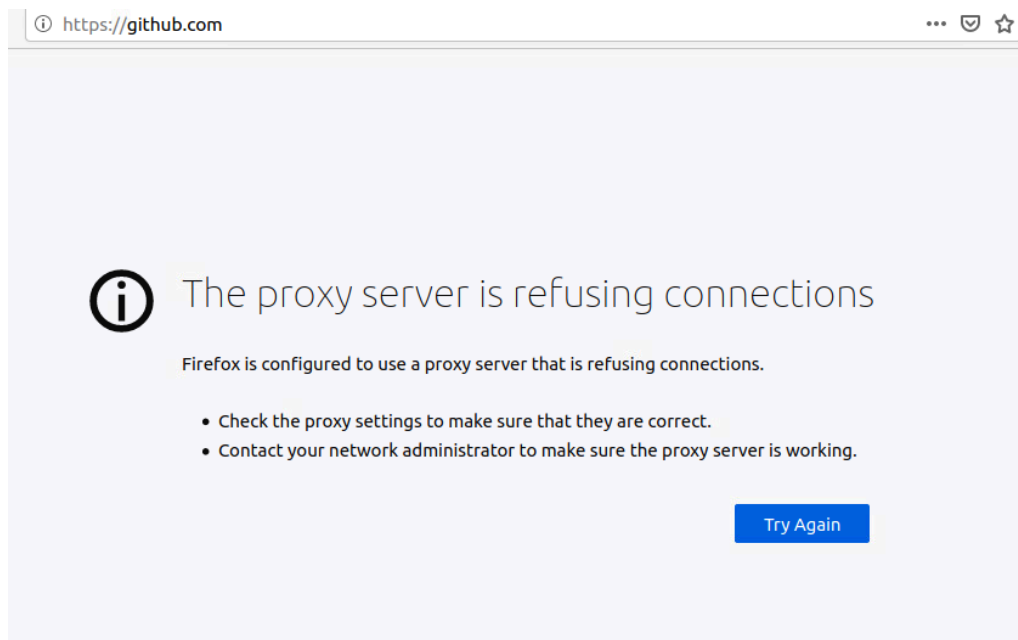
```
seed@10.8.0.99's password:
^Croot@aa5b1a15ad36:/#
```


7. Check if the connections still work - they don't, shows error of proxy not receiving packets for both websites:

a. example.com



b. github.com



Task 2.3 - Writing SOCKS Client Using Python

1. Change code to match proxy B and port:

```
1#!/bin/env python3
2
3import socks
4
5s = socks.socksocket()
6
7# Set the proxy
8s.set_proxy(socks.SOCKS5, "192.168.20.99", 7200)
9
10# Connect to final destination via the proxy
11hostname = "www.example.com"
12s.connect((hostname, 80))
13
14request = b"GET / HTTP/1.0\r\nHost: " + hostname.encode('utf-8') + b"\r\n\r\n"
15s.sendall(request)
16
17# Get the response
18response = s.recv(2048)
19while response:
20    print(response.split(b"\r\n"))
21    response = s.recv(2048)
```

2. Dynamic port forwarding using port B:

```
root@aa5b1a15ad36:/# ssh -4NT -D 192.168.20.99:7200 seed@10.8.0.99
seed@10.8.0.99's password:
```

3. Connect to proxy B1:

```
[11/17/24]seed@VM:~/.../Labsetup$ dockps
f9c53d8316e8  router-firewall
7d437c93369e  A1-10.8.0.5
ab137e711014  B1-192.168.20.5
8c9308f8e0e5  A-10.8.0.99
26b4da137414  B2-192.168.20.6
aa5b1a15ad36  B-192.168.20.99
5866db174db3  A2-10.8.0.6
[11/17/24]seed@VM:~/.../Labsetup$ docksh ab1
root@ab137e711014:/#
```

4. Add new file socksclient.py into the proxy directory (/home/seed) and run socksclient.py on B1 - runs successfully and shows html code:

```
root@ab137e711014:/home/seed# chmod 777 socksclient.py
root@ab137e711014:/home/seed# ./socksclient.py
[b'HTTP/1.0 200 OK', b'Age: 319653', b'Cache-Control: max-age=604800', b'Content-Type: text/html; charset=UTF-8', b'Date: Sun, 17 Nov 2024 19:22:19 GMT', b'Etag: "3147526947+ident"', b'Expires: Sun, 24 Nov 2024 19:22:19 GMT', b'Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT', b'Server: ECACC (chd/0755)', b'Vary: Accept-Encoding', b'X-Cache: HIT', b'Content-Length: 1256', b'Connection: close', b'', b'<!doctype html>\n<html>\n<head>\n    <title>Example Domain</title>\n\n    <meta charset="utf-8" />\n    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />\n    <meta name="viewport" content="width=device-width, initial-scale=1" />\n    <style type="text/css">\n        body {\n            background-color: #f0f0f2;\n            margin: 0;\n            padding: 0;\n            font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;\n        }\n        div {\n            width: 600px;\n            margin: 5em auto;\n            padding: 2em;\n            background-color: #fdfdff;\n            border-radius: 0.5em;\n            box-shadow: 2px 3px 7px 2px rgba(
```

5. Do the same steps for proxy B2 - gives back same result:

```
[11/17/24]seed@VM:~/.../Labsetup$ docksh 26b
root@26b4da137414:/# cd /home/seed
root@26b4da137414:/home/seed# ls
root@26b4da137414:/home/seed# nano socksclient.py
root@26b4da137414:/home/seed# chmod 777 socksclient.py
root@26b4da137414:/home/seed# ./socksclient.py
[b'HTTP/1.0 200 OK', b'Age: 399711', b'Cache-Control: max-age=604800', b'Content-Type: text/html; charset=UTF-8', b'Date: Sun, 17 Nov 2024 19:26:58 GMT', b'Etag: "3147526947+ident"', b'Expires: Sun, 24 Nov 2024 19:26:58 GMT', b'Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT', b'Server: ECACC (chd/071F)', b'Vary: Accept-Encoding', b'X-Cache: HIT', b'Content-Length: 1256', b'Connection: close', b'', b'<!doctype html>\n<html>\n<head>\n    <title>Example Domain</title>\n\n    <meta charset="utf-8" />\n    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />\n    <meta name="viewport" content="width=device-width, initial-scale=1" />\n    <style type="text/css">\n        body {\n            background-color: #f0f0f2;\n            margin: 0;\n            padding: 0;\n            font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;\n        }\n        div {\n            width: 600px;\n            margin: 5em auto;\n            padding: 2em;\n            background-color: #fdfdff;\n            border-radius: 0.5em;\n            box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);\n        }\n        a:link, a:visited {\n            color: #38488f;\n            text-decoration: none;\n        }\n        @media (max-width: 700px) {\n            div {\n                margin: 0 auto;\n                width: auto;\n            }\n        }\n    </style>\n</head>\n\n<body>\n<div>\n    <h1>Example Domain</h1>\n    <p>This doma
```

Task 3 - Connecting VPNs

Task 3.1 - Bypassing ingress (Internal to External) Firewall

1. Connect to Root A, write the command to create a tunnel from proxy A to proxy B

```
root@8c9308f8e0e5:/# ssh -w 0:0 root@192.168.20.99 -o "PermitLocalCommand=yes" -o "LocalCommand=ip addr add 192.168.53.88/24 dev tun0 && ip link set tun0 up" -o "RemoteCommand=ip addr add 192.168.53.99/24 dev tun0 && ip link set tun0 up"
The authenticity of host '192.168.20.99 (192.168.20.99)' can't be established.
ECDSA key fingerprint is SHA256:xo3ZDthadrlNI+uM+7gcsNVidtcdEaEyMezckwu8mbU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.20.99' (ECDSA) to the list of known hosts.
root@192.168.20.99's password:
```

2. Connect to host B1 and telnet to host A, you can see the seed connects to A at the end and Wireshark shows the Telnet packets between proxy:

```
root@ab137e711014:/# telnet 10.8.0.99
Trying 10.8.0.99...
Connected to 10.8.0.99.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
8c9308f8e0e5 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
seed@8c9308f8e0e5:~$
```

2024-11-17 14:55:10.80.99	192.168.20.5	TCP	68	TCP
7 2024-11-17 14:55:10.8.0.99	192.168.20.5	TCP	68	TCP
8 2024-11-17 14:55:10.8.0.99	192.168.20.5	TCP	68	TCP
9 2024-11-17 14:55:10.8.0.99	192.168.20.5	TELNET	72	Telnet
10 2024-11-17 14:55:10.8.0.99	192.168.20.5	TCP	72	TCP

* Frame 1: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface any,			
* Linux cooked capture			
* Internet Protocol Version 4, Src: 192.168.20.5, Dst: 10.8.0.99			
* Transmission Control Protocol, Src Port: 51336, Dst Port: 23, Seq: 990179742, Ack: 1			
* Telnet			

0000	00 03 00 01 00 00 02 42 c0 a8 14 05 00 00 00 00	...	0
0010	45 10 00 37 c2 04 40 00 40 06 78 c4 c0 a8 14 05	E:7	0	@.x.....
0020	0a 08 00 63 c8 88 00 17 3b 04 f1 9e 42 4e f8 c8	...	C	...;...BN...
0030	80 18 01 f6 df 41 00 00 01 01 08 6a 79 ff f9 42	...	A	...y..B
0040	0a 09 74 55 1b 5b 43	...	TU	[C

3. Connect to host B2 and telnet to host A1 - same result, connects to A1:

```
[11/17/24]seed@VM:~/../Labsetup$ docksh 26
root@26b4da137414:/# telnet 10.8.0.5
Trying 10.8.0.5...
Connected to 10.8.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
7d437c93369e login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

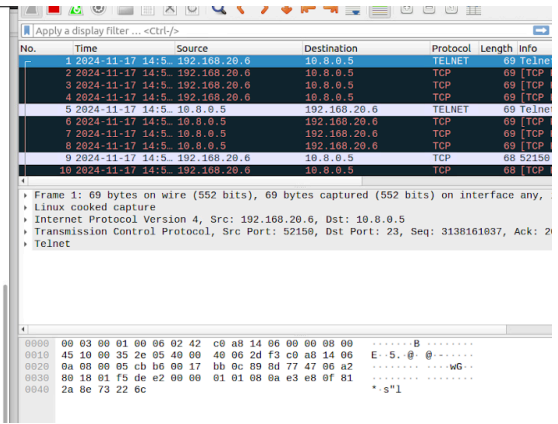
 * Documentation:  https://help.ubuntu.com
 * Management:   https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
```



4. Why can the packets go through:

The packets get through the firewall because they are hidden inside the secure tunnel (VPN). The firewall can see the packet coming through, but not its content, so it lets it pass through. The real information the packets is only visible to the sender and the receiver at the ends of the tunnel.

Task 3.2 - Bypassing Egress (External to Internal) Firewall

1. Create a tunnel from host B to A, local command and remote command would also be flipped:

```
^Cot@aa5b1a15ad36:/# ssh -w 0:0 root@10.8.0.99 -o "PermitLocalCommand=yes" -o "
LocalCommand=ip addr add 192.168.53.99/24 dev tun0 && ip link set tun0 up" -o "
RemoteCommand=ip addr add 192.168.53.88/24 dev tun0 && ip link set tun0 up"
root@10.8.0.99's password:
█
```

2. Connect to B2 and telnet to host A2 - successfully logs into seed A2:

```
[11/17/24]seed@VM:~/../Labsetup$ docksh 26
root@26b4da137414:/# telnet 10.8.0.6
Trying 10.8.0.6...
Connected to 10.8.0.6.
Escape character is '^'.
Ubuntu 20.04.1 LTS
5866db174db3 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

Time	Source	Destination	Protocol	Length	Info
1	2024-11-17 15:0	192.168.20.6	10.8.0.6	Telnet	69 Telnet D
2	2024-11-17 15:0	192.168.20.6	10.8.0.6	TCP	69 [TCP Ke
3	2024-11-17 15:0	192.168.20.6	10.8.0.6	TCP	69 [TCP Ke
4	2024-11-17 15:0	192.168.20.6	10.8.0.6	TCP	69 [TCP Ke
5	2024-11-17 15:0	10.8.0.6	192.168.20.6	TCP	68 23 - 509
6	2024-11-17 15:0	10.8.0.6	192.168.20.6	TCP	68 [TCP Ke
7	2024-11-17 15:0	10.8.0.6	192.168.20.6	TCP	68 [TCP Ke
8	2024-11-17 15:0	10.8.0.6	192.168.20.6	TCP	68 [TCP Ke
9	2024-11-17 15:0	10.8.0.6	192.168.20.6	Telnet	69 Telnet D
10	2024-11-17 15:0	10.8.0.6	192.168.20.6	TCP	69 [TCP Ke

Frame 11: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface any, id 0
Linux cooked capture
Internet Protocol Version 4, Src: 192.168.20.6, Dst: 10.8.0.6
Transmission Control Protocol, Src Port: 9996, Dst Port: 23, Seq: 3025453170, Ack: 74968
telnet

00 00 03 00 01 00 06 02 42 c0 a8 14 06 00 00 00 00B.....
10 45 10 00 35 0a 6a 40 00 40 06 d1 8c c0 a8 14 06 E..5.j0.0.....
20 0a 00 00 06 ea 34 00 17 b4 54 c0 72 2c af 55 a64...T.r..U..
30 00 18 01 f5 de a3 00 00 01 01 08 0a be c8 4f 621.....0b
40 5c f1 b7 9a 6c1

3. Exit A2 and connect to example.com through B2 (Successful):

```
Connection closed by foreign host.
root@26b4da137414:/# curl http://example.com
<!doctype html>
<html>
<head>
  <title>Example Domain</title>

  <meta charset="utf-8" />
  <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <style type="text/css">
    body {
      background-color: #f0f0f2;
      margin: 0;
      padding: 0;
      font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI",
'Open Sans', "Helvetica Neue", Helvetica, Arial, sans-serif;

    }
    div {
      width: 600px;
      margin: 5em auto;
      padding: 2em;
      background-color: #f0f0f2.
```

4. Why do the packets go through:

The VPN tunnel encrypts the traffic between 192.168.53.88 (client) and 192.168.53.99 (server). The firewall can only see encrypted packets traveling through the tunnel and doesn't know their true destination. The NAT rule changes the source IP of the packets leaving Machine A from 192.168.53.88 (client) to 10.8.0.99 (server). Since the firewall allows traffic from 10.8.0.99, it lets the packets pass through. The VPN disguises the blocked traffic as allowed traffic by encrypting it and using the VPN server's permitted IP address for outgoing packets. This allows it to bypass the firewall restrictions effectively.

Task 4: Comparing SOCKs and VPNs:

1. General purposes and uses:

SOCKS5 proxies and VPNs are tools used to get around firewalls, but they work in different ways and are useful for different situations. Think of a SOCKS5 proxy as a middleman for specific apps, like your web browser. When you use it, the proxy sends your app’s traffic to the internet on your behalf. It’s simple to set up and was shown in your lab when you used dynamic port forwarding to access blocked websites. However, the proxy doesn’t hide what your traffic is or protect it—it just changes where it looks like the traffic is coming from. This means that anyone monitoring the connection, like a firewall or attacker, can still see what you’re sending and receiving.

A VPN, on the other hand, is like a secure tunnel that protects everything coming from your computer. It encrypts all your traffic, so even if someone is monitoring the connection, they can’t see what you’re doing. In your lab, the VPN not only bypassed the firewall but also made sure the data was hidden from prying eyes by encrypting it. While VPNs are a bit more complicated to set up and might slow things down slightly because of the extra processing needed for encryption, they provide much stronger security and privacy. Essentially, a SOCKS5 proxy is quick and simple for specific tasks, while a VPN is a more powerful tool for protecting all your internet traffic and bypassing firewalls completely.

2. Differences, pros and cons:

a. Difference

Feature	SOCKS5	VPN
Traffic Encryption	No encryption; data is visible to anyone monitoring the connection.	Fully encrypts all traffic, ensuring privacy and security.
Scope of Protection	Protects only the traffic from apps configured to use the proxy.	Protects all traffic from the device, regardless of the application.
Firewall Bypassing	Effective for bypassing app-specific restrictions by rerouting traffic through the proxy.	Fully bypasses firewalls by encrypting and hiding all traffic.
Privacy and Security	Limited; does not hide your IP or encrypt data, leaving it exposed to eavesdropping.	High; encrypts data and hides your IP, ensuring privacy and security.

Use Case	Best for simple tasks like browsing blocked websites or accessing region-specific content.	Best for securing all internet traffic, bypassing restrictions, and ensuring privacy.
Pros	<ul style="list-style-type: none"> - Easy to set up and use. - Faster since there's no encryption overhead. 	<ul style="list-style-type: none"> - Encrypts all traffic, ensuring privacy. - Hides your IP, protecting your identity as not to get caught when bypassing the firewall
Cons	<ul style="list-style-type: none"> - No encryption, so data isn't secure. - Only works for configured apps. 	<ul style="list-style-type: none"> - Slightly more complex to set up in real world. - Can slow down connection due to encryption processing.