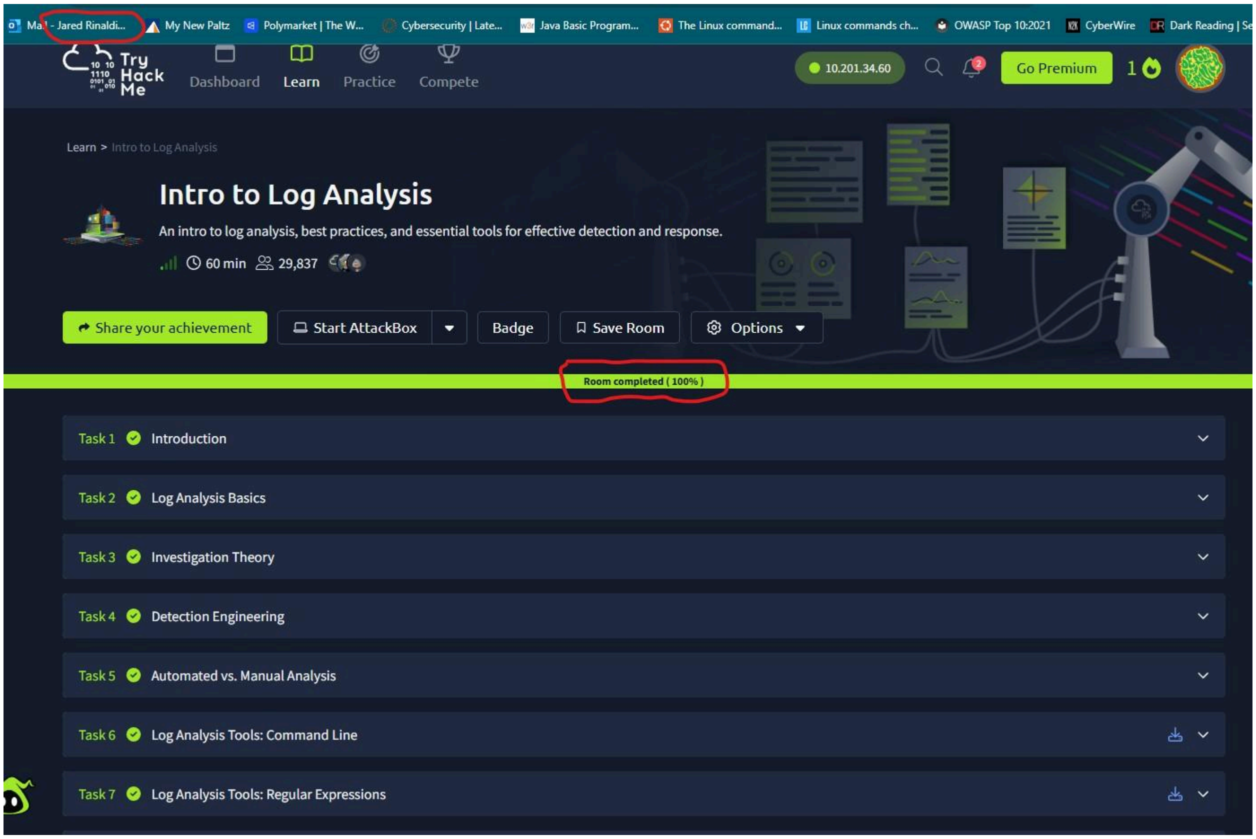
Configuring an Uncomplicated Firewall

Researcher: Jared Rinaldi
Github.com/PyronicGreen
December 2, 2025

Abstract: In this lab, I will configure an uncomplicated firewall (UFW) on my Ubuntu VM. UFW is Ubuntu's default command-line tool for managing firewall rules. It provides a straightforward interface for creating and maintaining a host-based firewall that supports both IPv4 or IPv6 traffic. Rather than replacing the underlying Linux firewall system, UFW serves as a frontend tool that applies rules through `iptables` (or nftables on newer systems). It is designed to make secure configurations easy by enabling effective defaults, requiring only a few straightforward commands to allow or deny network connections.

TryHackMe:



Learn > Intro to Log Analysis

Intro to Log Analysis

An intro to log analysis, best practices, and essential tools for effective detection and response.

60 min 29,837

Room completed (100%)

Task 1 Introduction

Task 2 Log Analysis Basics

Task 3 Investigation Theory

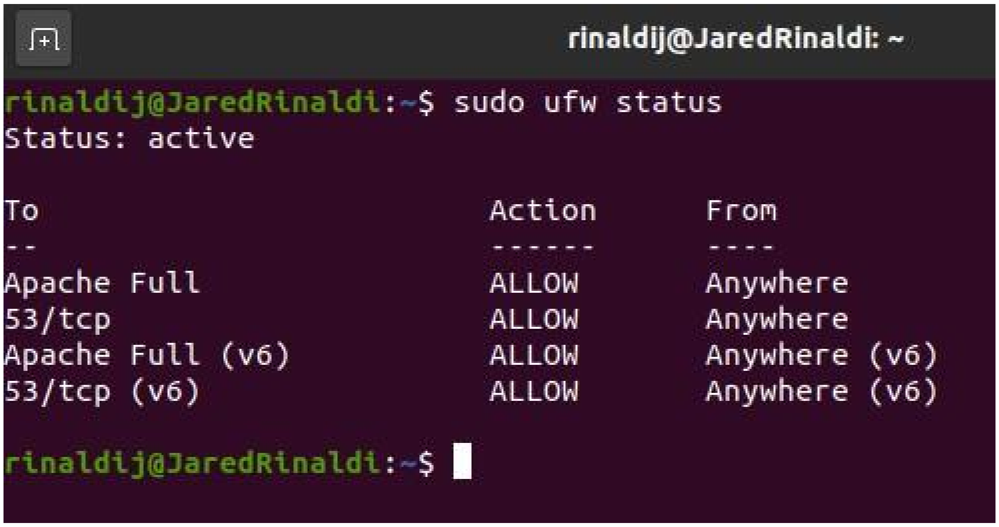
Task 4 Detection Engineering

Task 5 Automated vs. Manual Analysis

Task 6 Log Analysis Tools: Command Line

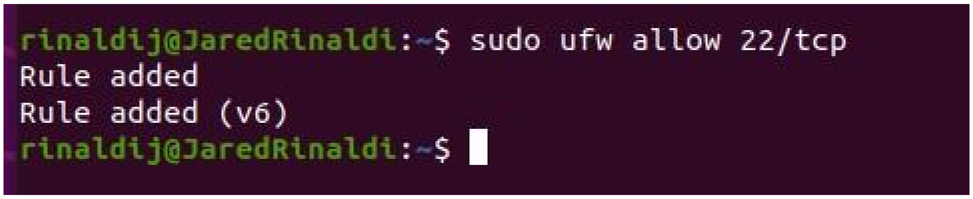
Task 7 Log Analysis Tools: Regular Expressions

I. Enable UFW:

1. 

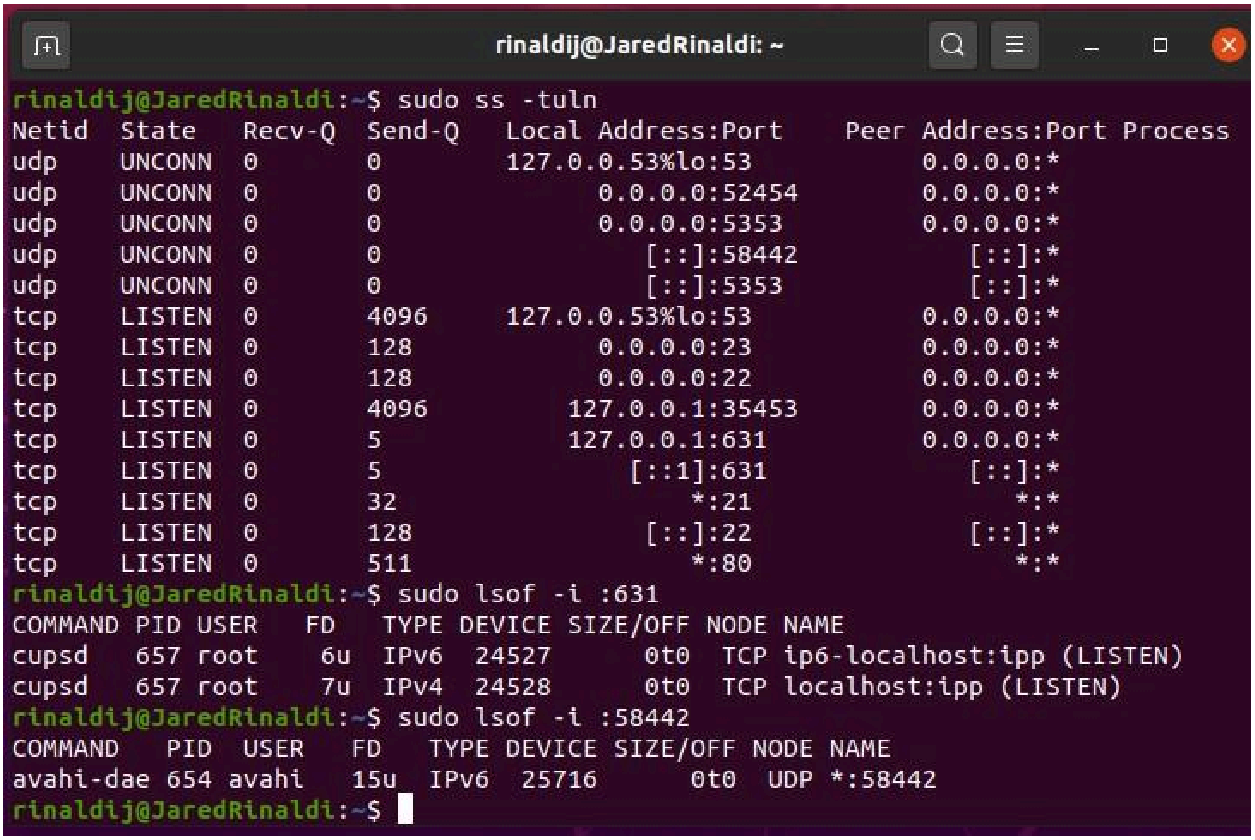
```
rinaldij@JaredRinaldi:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Apache Full                ALLOW       Anywhere
53/tcp                     ALLOW       Anywhere
Apache Full (v6)           ALLOW       Anywhere (v6)
53/tcp (v6)                ALLOW       Anywhere (v6)

rinaldij@JaredRinaldi:~$
```

Prior to this lab, I read through the baeldung article that was assigned and performed some config changes on the UFW, including enabling Apache and DNS for TCP. This is why the output looks the way it does for question 1.

2. 

```
rinaldij@JaredRinaldi:~$ sudo ufw allow 22/tcp
Rule added
Rule added (v6)
rinaldij@JaredRinaldi:~$
```

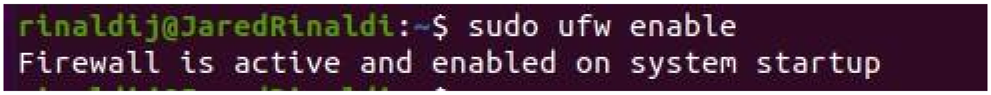
The reason that traffic should be permitted through port 22 (SSH) prior to enabling the UFW is that the UFW will block all traffic by default, so if outgoing traffic through port 22 isn't enabled, the server will not be able to be accessed.

3. 

```
rinaldij@JaredRinaldi:~$ sudo ss -tuln
Netid  State   Recv-Q  Send-Q   Local Address:Port    Peer Address:Port Process
udp    UNCONN  0       0        127.0.0.53%lo:53           0.0.0.0:*
udp    UNCONN  0       0              0.0.0.0:52454        0.0.0.0:*
udp    UNCONN  0       0              0.0.0.0:5353         0.0.0.0:*
udp    UNCONN  0       0                 [::]:58442           [::]:*
udp    UNCONN  0       0                 [::]:5353            [::]:*
tcp    LISTEN  0       4096     127.0.0.53%lo:53           0.0.0.0:*
tcp    LISTEN  0       128            0.0.0.0:23           0.0.0.0:*
tcp    LISTEN  0       128            0.0.0.0:22           0.0.0.0:*
tcp    LISTEN  0       4096         127.0.0.1:35453        0.0.0.0:*
tcp    LISTEN  0       5            127.0.0.1:631          0.0.0.0:*
tcp    LISTEN  0       5                [::1]:631             [::]:*
tcp    LISTEN  0       32                   *:21                 *:*
tcp    LISTEN  0       128               [::]:22              [::]:*
tcp    LISTEN  0       511                  *:80                 *:*
rinaldij@JaredRinaldi:~$ sudo lsof -i :631
COMMAND PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
cupsd   657 root    6u  IPv6  24527      0t0  TCP ip6-localhost:ipp (LISTEN)
cupsd   657 root    7u  IPv4  24528      0t0  TCP localhost:ipp (LISTEN)
rinaldij@JaredRinaldi:~$ sudo lsof -i :58442
COMMAND   PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
avahi-dae 654 avahi   15u  IPv6  25716      0t0  UDP *:58442
rinaldij@JaredRinaldi:~$
```

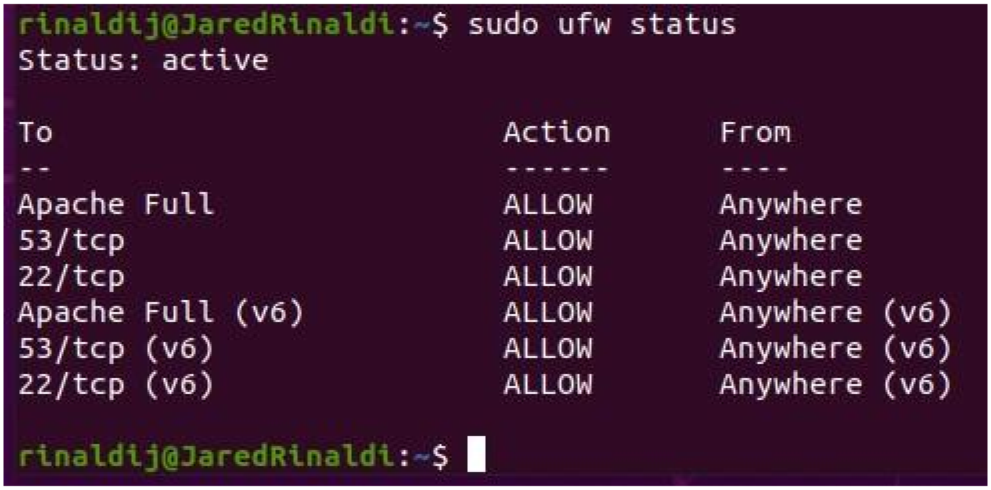
I wasn't sure what services ports 631 and 58442 provided, so I ran the lsof command to get more information about them.

- **631 - CUPS**: cupsd is a type of printer scheduler for the system. CUPS stands for Common Unit (or UNIX) Printing System.
- **58442 - avahi-dae**: The avahi daemon is a Linux service whose aim is to let devices that are connected to the local network broadcast their IP address together with their function.

4. 

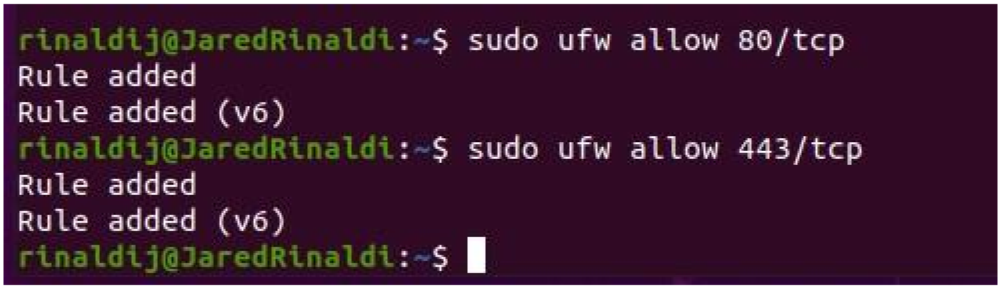
```
rinaldij@JaredRinaldi:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

5.



```
rinaldij@JaredRinaldi:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Apache Full                ALLOW       Anywhere
53/tcp                     ALLOW       Anywhere
22/tcp                     ALLOW       Anywhere
Apache Full (v6)           ALLOW       Anywhere (v6)
53/tcp (v6)                ALLOW       Anywhere (v6)
22/tcp (v6)                ALLOW       Anywhere (v6)

rinaldij@JaredRinaldi:~$
```

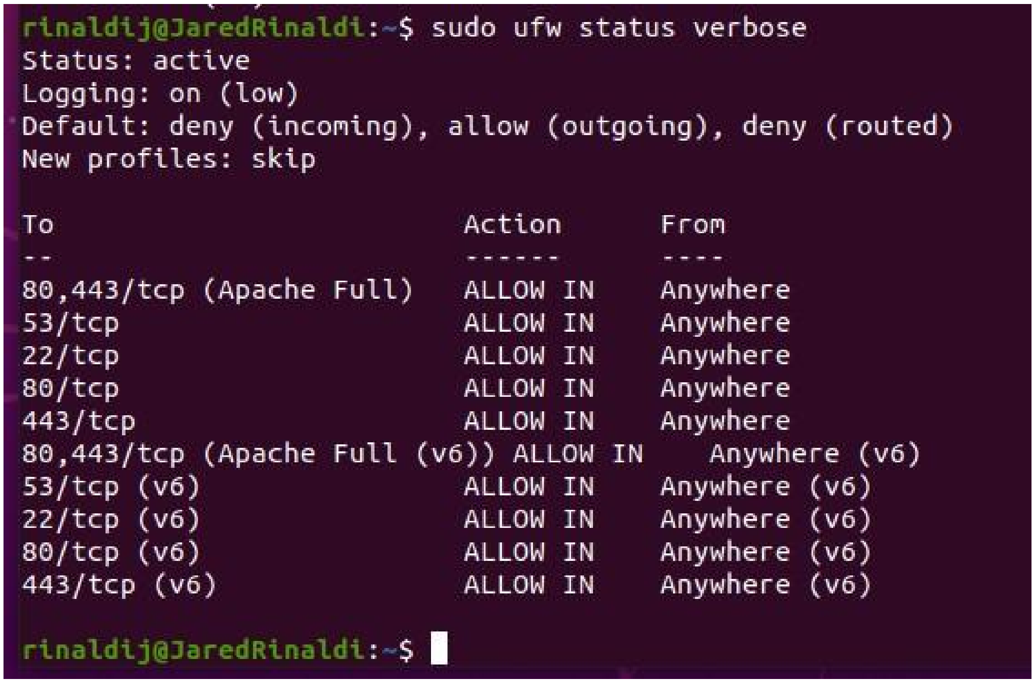
6.



```
rinaldij@JaredRinaldi:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
rinaldij@JaredRinaldi:~$ sudo ufw allow 443/tcp
Rule added
Rule added (v6)
rinaldij@JaredRinaldi:~$
```

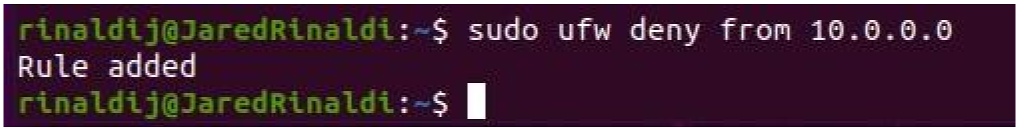
The ports that should be allowed in UFW are 80 for HTTP and 443 for HTTPS.

7.

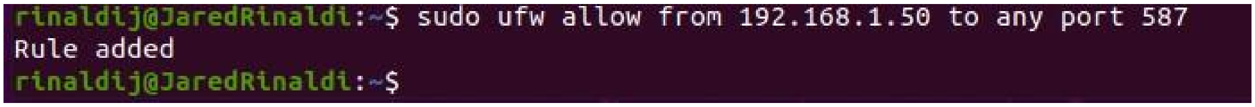


```
rinaldij@JaredRinaldi:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
80,443/tcp (Apache Full)   ALLOW IN    Anywhere
53/tcp                     ALLOW IN    Anywhere
22/tcp                     ALLOW IN    Anywhere
80/tcp                     ALLOW IN    Anywhere
443/tcp                    ALLOW IN    Anywhere
80,443/tcp (Apache Full (v6)) ALLOW IN    Anywhere (v6)
53/tcp (v6)                ALLOW IN    Anywhere (v6)
22/tcp (v6)                ALLOW IN    Anywhere (v6)
80/tcp (v6)                ALLOW IN    Anywhere (v6)
443/tcp (v6)               ALLOW IN    Anywhere (v6)

rinaldij@JaredRinaldi:~$
```

There is a good amount of information that looks useful here. The default settings are listed at the top of the output. It is also useful to see the type of IP being used (IPv4 or IPv6) as well as the type of connection, which is TCP in all the services.

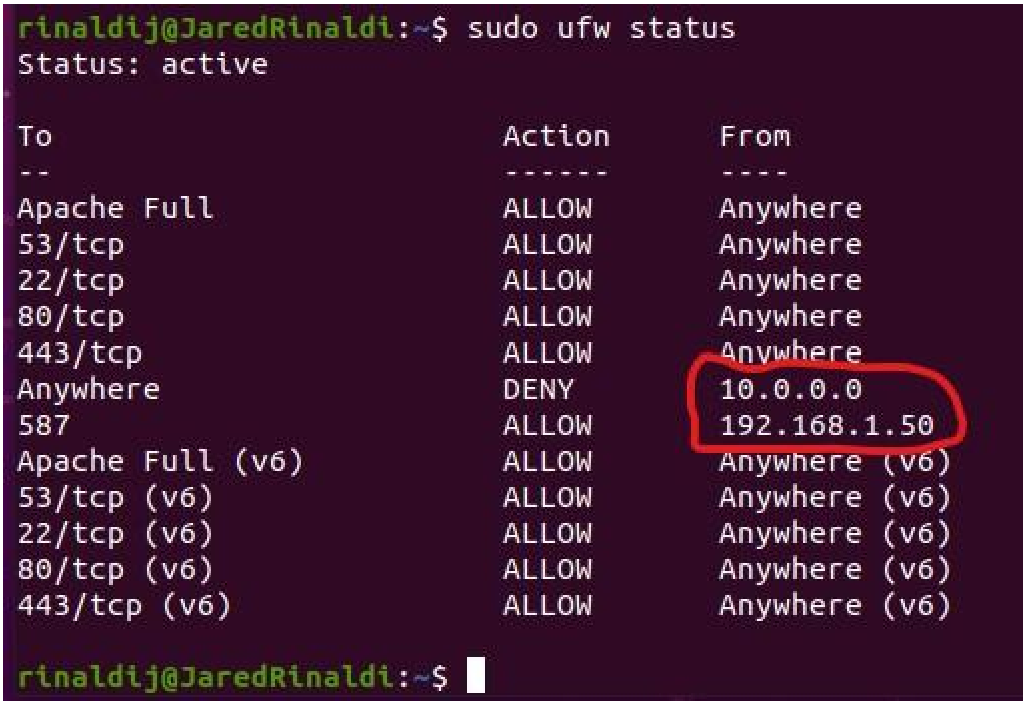
8. 

```
rinaldij@JaredRinaldi:~$ sudo ufw deny from 10.0.0.0
Rule added
rinaldij@JaredRinaldi:~$
```

9. 

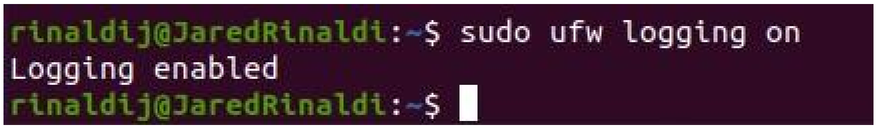
```
rinaldij@JaredRinaldi:~$ sudo ufw allow from 192.168.1.50 to any port 587
Rule added
rinaldij@JaredRinaldi:~$
```

Port 587 is typically used for SMTP, which is the Simple Mail Transfer Protocol used for email exchange. While port 465 used to be the standard for secure email transmission, it is now less commonly used compared to port 587. This is because port 587 supports **explicit TLS** (often referred to as STARTTLS), which can make insecure connections secure using TLS (Transport Layer Security).

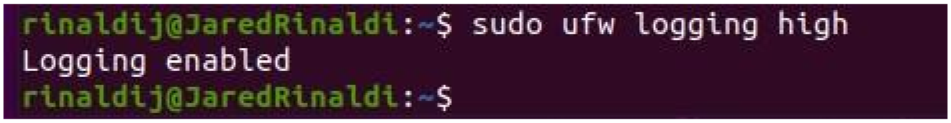
10. 

```
rinaldij@JaredRinaldi:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Apache Full                ALLOW       Anywhere
53/tcp                     ALLOW       Anywhere
22/tcp                     ALLOW       Anywhere
80/tcp                     ALLOW       Anywhere
443/tcp                    ALLOW       Anywhere
Anywhere                   DENY        10.0.0.0
587                        ALLOW       192.168.1.50
Apache Full (v6)           ALLOW       Anywhere (v6)
53/tcp (v6)                ALLOW       Anywhere (v6)
22/tcp (v6)                ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)
443/tcp (v6)               ALLOW       Anywhere (v6)

rinaldij@JaredRinaldi:~$
```

II. Enable UFW Logging

1. 

```
rinaldij@JaredRinaldi:~$ sudo ufw logging on
Logging enabled
rinaldij@JaredRinaldi:~$
```

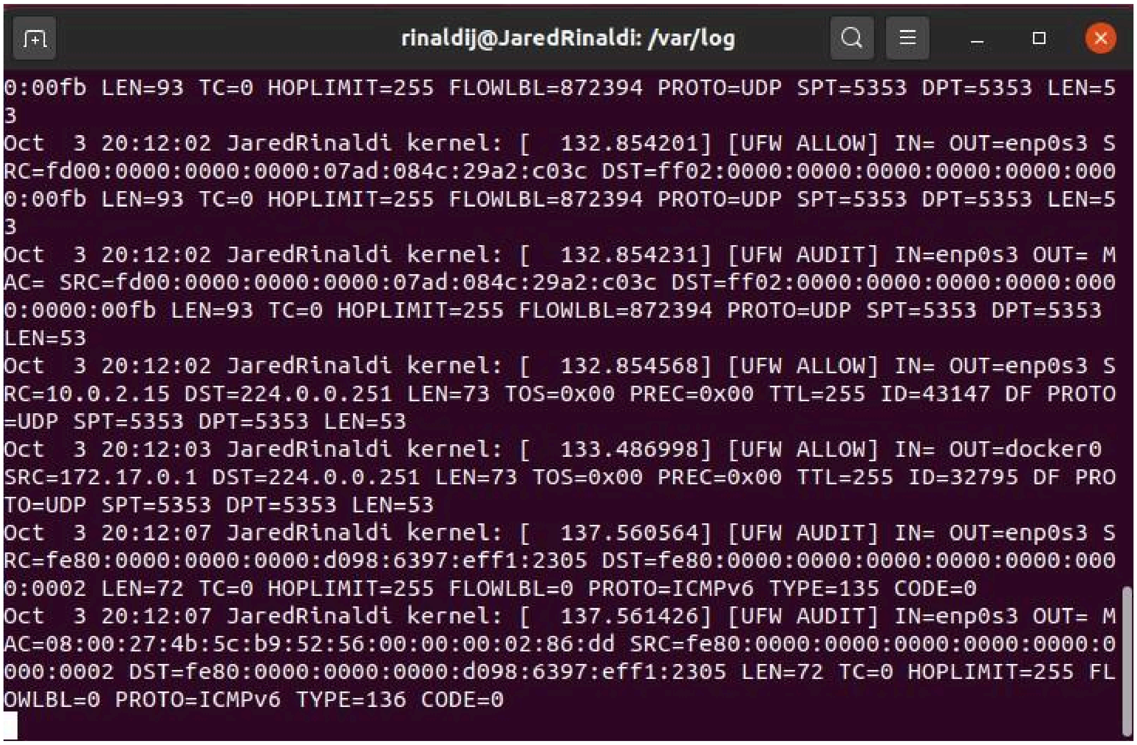
2. 

```
rinaldij@JaredRinaldi:~$ sudo ufw logging high
Logging enabled
rinaldij@JaredRinaldi:~$
```

3. The information in a UFW log entry is useful because it allows you to better identify what traffic is allowed or not allowed into your machine. It allows for better port security and can make you aware of any malicious actors that may be in or attempting to break into your system.
    a. **MAC Address**: This is the number associated with the physical machine that was the source of the Ethernet frame.
    b. **SRC**: Source IP address of the traffic. This is the sender's IP address.
    c. **DST**: Destination IP address. This is the recipient of the traffic.
    d. **SPT**: Source port. This is the port number on the client. It is often randomly assigned (ephemeral).
    e. **DPT**: This is the destination port on the destination server.
    f. **PROTO**: Protocol describes what transport protocol the packet is using. This will often be TCP or UDP.
    g. **UFW BLOCK**: This will indicate that a packet was blocked by UFW.

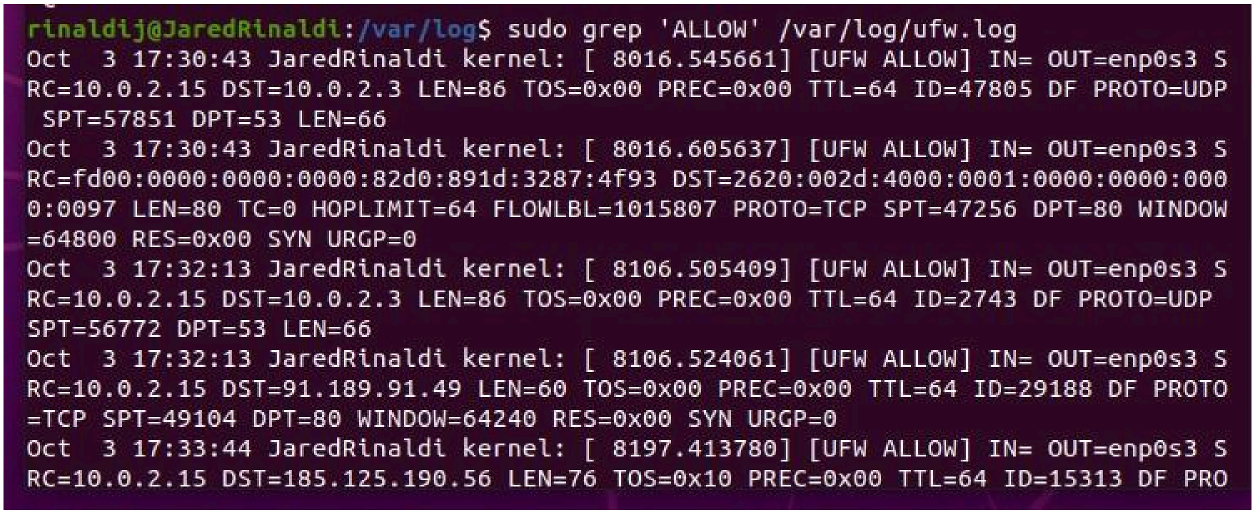
4. Monitoring UFW logs in real time after running command:

```
Sudo tail -f /var/log/ufw.log
```



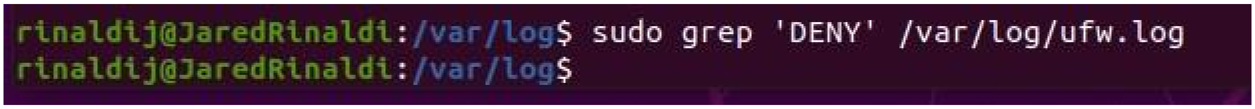
```
0:00fb LEN=93 TC=0 HOPLIMIT=255 FLOWLBL=872394 PROTO=UDP SPT=5353 DPT=5353 LEN=5
3
Oct  3 20:12:02 JaredRinaldi kernel: [  132.854201] [UFW ALLOW] IN= OUT=enp0s3 S
RC=fd00:0000:0000:0000:07ad:084c:29a2:c03c DST=ff02:0000:0000:0000:0000:0000:000
0:00fb LEN=93 TC=0 HOPLIMIT=255 FLOWLBL=872394 PROTO=UDP SPT=5353 DPT=5353 LEN=5
3
Oct  3 20:12:02 JaredRinaldi kernel: [  132.854231] [UFW AUDIT] IN=enp0s3 OUT= M
AC= SRC=fd00:0000:0000:0000:07ad:084c:29a2:c03c DST=ff02:0000:0000:0000:0000:000
0:0000:00fb LEN=93 TC=0 HOPLIMIT=255 FLOWLBL=872394 PROTO=UDP SPT=5353 DPT=5353
LEN=53
Oct  3 20:12:02 JaredRinaldi kernel: [  132.854568] [UFW ALLOW] IN= OUT=enp0s3 S
RC=10.0.2.15 DST=224.0.0.251 LEN=73 TOS=0x00 PREC=0x00 TTL=255 ID=43147 DF PROTO
=UDP SPT=5353 DPT=5353 LEN=53
Oct  3 20:12:03 JaredRinaldi kernel: [  133.486998] [UFW ALLOW] IN= OUT=docker0
SRC=172.17.0.1 DST=224.0.0.251 LEN=73 TOS=0x00 PREC=0x00 TTL=255 ID=32795 DF PRO
TO=UDP SPT=5353 DPT=5353 LEN=53
Oct  3 20:12:07 JaredRinaldi kernel: [  137.560564] [UFW AUDIT] IN= OUT=enp0s3 S
RC=fe80:0000:0000:0000:d098:6397:eff1:2305 DST=fe80:0000:0000:0000:0000:0000:000
0:0002 LEN=72 TC=0 HOPLIMIT=255 FLOWLBL=0 PROTO=ICMPv6 TYPE=135 CODE=0
Oct  3 20:12:07 JaredRinaldi kernel: [  137.561426] [UFW AUDIT] IN=enp0s3 OUT= M
AC=08:00:27:4b:5c:b9:52:56:00:00:00:02:86:dd SRC=fe80:0000:0000:0000:0000:0000:0
000:0002 DST=fe80:0000:0000:0000:d098:6397:eff1:2305 LEN=72 TC=0 HOPLIMIT=255 FL
OWLBL=0 PROTO=ICMPv6 TYPE=136 CODE=0
```

5. Allowed traffic:



```
rinaldij@JaredRinaldi:/var/log$ sudo grep 'ALLOW' /var/log/ufw.log
Oct  3 17:30:43 JaredRinaldi kernel: [ 8016.545661] [UFW ALLOW] IN= OUT=enp0s3 S
RC=10.0.2.15 DST=10.0.2.3 LEN=86 TOS=0x00 PREC=0x00 TTL=64 ID=47805 DF PROTO=UDP
 SPT=57851 DPT=53 LEN=66
Oct  3 17:30:43 JaredRinaldi kernel: [ 8016.605637] [UFW ALLOW] IN= OUT=enp0s3 S
RC=fd00:0000:0000:0000:82d0:891d:3287:4f93 DST=2620:002d:4000:0001:0000:0000:000
0:0097 LEN=80 TC=0 HOPLIMIT=64 FLOWLBL=1015807 PROTO=TCP SPT=47256 DPT=80 WINDOW
=64800 RES=0x00 SYN URGP=0
Oct  3 17:32:13 JaredRinaldi kernel: [ 8106.505409] [UFW ALLOW] IN= OUT=enp0s3 S
RC=10.0.2.15 DST=10.0.2.3 LEN=86 TOS=0x00 PREC=0x00 TTL=64 ID=2743 DF PROTO=UDP
SPT=56772 DPT=53 LEN=66
Oct  3 17:32:13 JaredRinaldi kernel: [ 8106.524061] [UFW ALLOW] IN= OUT=enp0s3 S
RC=10.0.2.15 DST=91.189.91.49 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=29188 DF PROTO
=TCP SPT=49104 DPT=80 WINDOW=64240 RES=0x00 SYN URGP=0
Oct  3 17:33:44 JaredRinaldi kernel: [ 8197.413780] [UFW ALLOW] IN= OUT=enp0s3 S
RC=10.0.2.15 DST=185.125.190.56 LEN=76 TOS=0x10 PREC=0x00 TTL=64 ID=15313 DF PRO
```

Denied traffic:



```
rinaldij@JaredRinaldi:/var/log$ sudo grep 'DENY' /var/log/ufw.log
rinaldij@JaredRinaldi:/var/log$
```

There is **not** any denied traffic. This is due to there being no inbound traffic from external sources for my server to deny/block. I believe that running my server in a virtual machine (Virtual Box, in this case) has the effect of blocking inbound traffic from reaching the VM. This is due to the host machine handling real inbound traffic.

Conclusion: In this lab, I was able to use UFW to allow and deny traffic onto my virtual machine's network. I was also able to log the traffic and see it populate in my terminal in real-time.