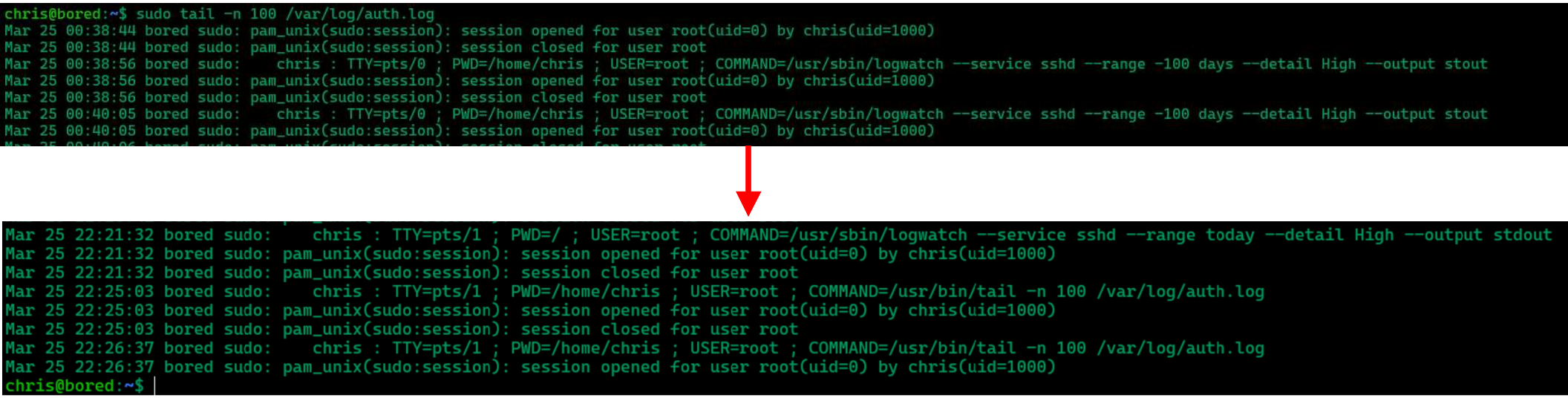
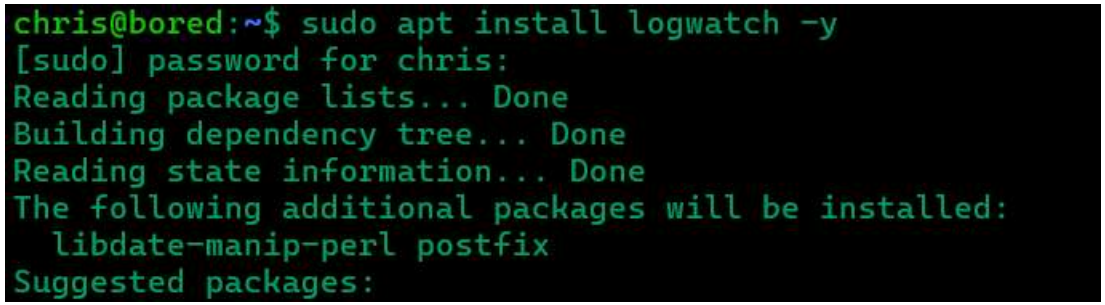
Logwatch into a tcpdump...enjoy the ride!

Lets start and check the last 100 lines of the “auth.log”. There are multiple ways to accomplish the mission but to read the **last** part tells us “tail” and 100 lines means “-n 100”. Remember, default tail length is 10 lines.



Looking through auth.log is possible and we can understand it but is there another way? Of course, so lets go ahead and install Logwatch.



Now, lets do a quick check at SSH logins today using Logwatch. Oh, well my VM stays up most days and I stay remoted in. Makes sense that no returns. I think I remoted in yesterday so lets make a quick change and see.

```
chris@bored:~$ sudo logwatch --service sshd --detail High --range 'Today' --output stdout
chris@bored:~$ sudo logwatch --service sshd --detail High --range 'Yesterday' --output stdout

 ################### Logwatch 7.5.6 (07/23/21) ####################
        Processing Initiated: Wed Mar 20 03:03:23 2024
        Date Range Processed: yesterday
                              ( 2024-Mar-19 )
                              Period is day.
        Detail Level of Output: 10
        Type of Output/Format: stdout / text
        Logfiles for Host: bored
 ##################################################################

 --------------------- SSHD Begin ------------------------

 SSHD Started: 4 Times

 Users logging in through sshd:
    chris:
       10.0.0.170: 1 Time

 ---------------------- SSHD End -------------------------


 ###################### Logwatch End #########################
```

Results look a little easier to read than the auth.log version.

We now know how to see the SSH service but what else? Lets try “sudo”. Maybe someone is elevating privileges.

Nice job, also we see that I had an issue with spelling. Can you find it?

Reading that you know what I meant but computers need us to be specific or else we will not get correct results

```
chris@bored:~$ sudo logwatch --service sudo --range today --detail High --output stdout

 ################### Logwatch 7.5.6 (07/23/21) ####################
        Processing Initiated: Mon Mar 25 22:47:25 2024
        Date Range Processed: today
                              ( 2024-Mar-25 )
                              Period is day.
        Detail Level of Output: 10
        Type of Output/Format: stdout / text
        Logfiles for Host: bored
 ##################################################################

 --------------------- Sudo (secure-log) Begin ------------------------

 chris => root
 -------------
 /usr/bin/less /var/log/auth.log
 /usr/bin/tail -n 100 /var/log/auth.log
 /usr/sbin/logwatch --service sshd --range today --detail High --output \
     stout
 /usr/sbin/logwatch --service sshd --range yesterday --detail High --output \
     stout
 /usr/sbin/logwatch --service sshd --range -10 days --detail High --output \
     stout
 /usr/sbin/logwatch --service sshd --range -100 days --detail High --output \
     stout
 /usr/sbin/logwatch --service sshd --range -100 days --detail High --output \
     stout
 /usr/sbin/logwatch --service sshd --detail High --range -100 days --output \
     stdout
 /usr/sbin/logwatch --service sshd --range -100 days --detail High --output \
     stdout
 /usr/sbin/logwatch --service sshd --range -10 days --detail High --output \
     stdout
 /usr/sbin/logwatch --service sshd --range yesterday --detail High --output \
     stdout
 /usr/bin/apt install logwatch -y
 /usr/bin/apt-get update
 /usr/bin/find -name logwatch.*
 /usr/bin/mkdir /var/cache/logwatch
 /usr/sbin/logwatch --service sshd --range today --detail High --output \
     stdout
 /usr/bin/tail -n 100 /var/log/auth.log
 /usr/bin/tail -n 100 /var/log/auth.log
 /usr/sbin/logwatch --service All --range today --detail High --output \
     stdout
 /usr/sbin/logwatch --service sudo --range today --detail High --output \
     stdout
```

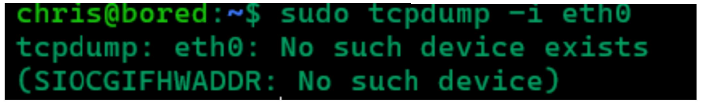
Logwatch is a helpful tool and there is more that can be done. I find logwatch helpful to quickly search and depending shift into another way to continue investigating.

Test it out with other services, date range, detail levels, and more. Find what works for you and you like.

Lets now take a look at tcpdump which is a tool that captures packets.

Simple test will be to use the command “tcpdump -i eth0”...did that work for you?

No luck for me but why?



Well, quick search will explain eth0 is the ethernet adapter/connection. Though what if I’m not using it? What can we do to troubleshoot?

Lets check our ip connection using “ifconfig” or “ip a”
Oh, looks like my adapter in use for my VM is enp0s3.

```
chris@bored:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:2a:8d:e9 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.161/24 metric 100 brd 10.0.0.255 scope global dynamic enp0s3
       valid_lft 164035sec preferred_lft 164035sec
    inet6 2601:281:d87e:6aa0::aac/128 scope global dynamic noprefixroute
       valid_lft 5228sec preferred_lft 5228sec
    inet6 2601:281:d87e:6aa0:a00:27ff:fe2a:8de9/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 300sec preferred_lft 300sec
    inet6 fe80::a00:27ff:fe2a:8de9/64 scope link
       valid_lft forever preferred_lft forever
```

Ok lets try this again but using enp0s3

```
chris@bored:~$ sudo tcpdump -i enp0s3
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
18:24:36.855121 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 4124536454:4124536650, ack 135527
0661, win 501, length 196
18:24:36.894620 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 196, win 8192, length 0
18:24:36.941917 IP bored.36189 > cdns01.comcast.net.domain: 41212+ [1au] PTR? 170.0.0.10.in-addr.a
rpa. (52)
```

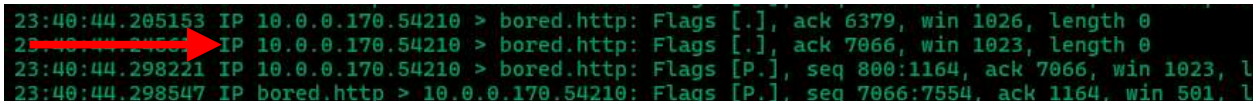
Success!

Lets grab some http packets next. You may not see any traffic initially. I suggest after you turn listening on for port 80, opening a browser and visit your web server. As soon as you go it will start capturing the traffic.

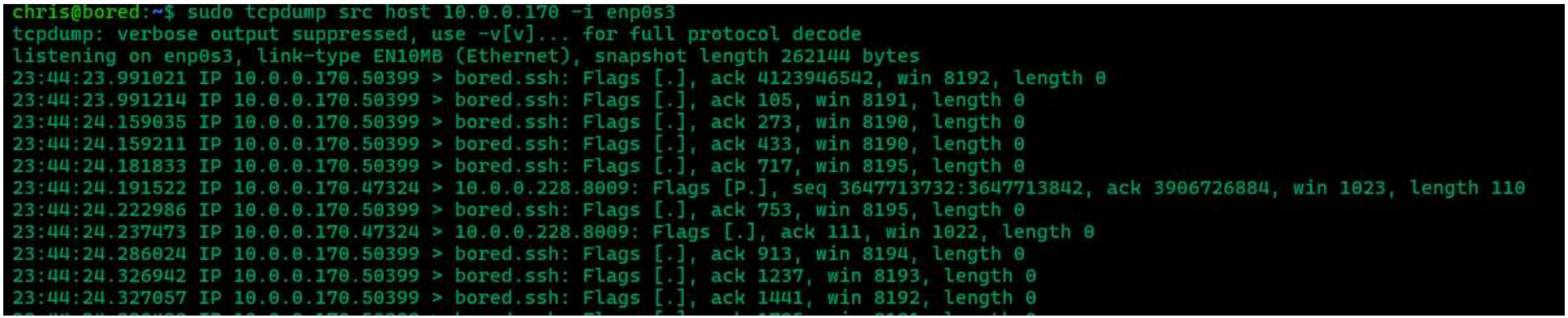
```
chris@bored:~$ sudo tcpdump port 80 -i enp0s3
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```


```
23:40:44.176222 IP 10.0.0.170.54210 > bored.http: Flags [S], seq 1798124037, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
23:40:44.176256 IP bored.http > 10.0.0.170.54210: Flags [S.], seq 3710823882, ack 1798124038, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
23:40:44.176505 IP 10.0.0.170.54210 > bored.http: Flags [.], ack 1, win 1026, length 0
23:40:44.176811 IP 10.0.0.170.54210 > bored.http: Flags [P.], seq 1:426, ack 1, win 1026, length 425: HTTP: GET / HTTP/1.1
23:40:44.176842 IP bored.http > 10.0.0.170.54210: Flags [.], ack 426, win 501, length 0
23:40:44.178462 IP bored.http > 10.0.0.170.54210: Flags [P.], seq 1:3459, ack 426, win 501, length 3458: HTTP: HTTP/1.1 200 OK
23:40:44.178893 IP 10.0.0.170.54210 > bored.http: Flags [.], ack 2921, win 1026, length 0
23:40:44.183328 IP 10.0.0.170.54211 > bored.http: Flags [S], seq 2221669650, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
23:40:44.183358 IP bored.http > 10.0.0.170.54211: Flags [S.], seq 4169775558, ack 2221669651, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
23:40:44.183596 IP 10.0.0.170.54211 > bored.http: Flags [.], ack 1, win 1026, length 0
23:40:44.203813 IP 10.0.0.170.54210 > bored.http: Flags [P.], seq 426:800, ack 3459, win 1024, length 374: HTTP: GET /icons/ubuntu-logo.png HTTP/1.1
23:40:44.204680 IP bored.http > 10.0.0.170.54210: Flags [P.], seq 3459:7066, ack 800, win 501, length 3607: HTTP: HTTP/1.1 200 OK
23:40:44.205153 IP 10.0.0.170.54210 > bored.http: Flags [.], ack 6379, win 1026, length 0
23:40:44.245614 IP 10.0.0.170.54210 > bored.http: Flags [.], ack 7066, win 1023, length 0
23:40:44.298221 IP 10.0.0.170.54210 > bored.http: Flags [P.], seq 800:1164, ack 7066, win 1023, length 364: HTTP: GET /favicon.ico HTTP/1.1
23:40:44.298547 IP bored.http > 10.0.0.170.54210: Flags [P.], seq 7066:7554, ack 1164, win 501, length 488: HTTP: HTTP/1.1 404 Not Found
23:40:44.339110 IP 10.0.0.170.54210 > bored.http: Flags [.], ack 7554, win 1021, length 0
23:40:49.300249 IP bored.http > 10.0.0.170.54210: Flags [F.], seq 7554, ack 1164, win 501, length 0
23:40:49.300444 IP 10.0.0.170.54210 > bored.http: Flags [.], ack 7555, win 1021, length 0
23:41:15.714253 IP bored.http > 10.0.0.170.54211: Flags [S.], seq 4169775558, ack 2221669651, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
```

Using the http capture we see the ip of the source. Lets see what they are up to and capture their packets specifically next.



We will tell it this time to target the source ip 10.0.0.170



Next lets packet capture against tcp traffic. The output hould be no surprise since its just us accessing our VM.

```
chris@bored:~$ sudo tcpdump tcp -i enp0s3
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
23:45:26.611114 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 4123957986:4123958094, ack 1355111057, win 501, length 108
23:45:26.611214 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 108:272, ack 1, win 501, length 164
23:45:26.611298 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 272, win 8194, length 0
23:45:26.611344 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 272:340, ack 1, win 501, length 68
23:45:26.611430 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 340:376, ack 1, win 501, length 36
23:45:26.611544 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 376, win 8193, length 0
23:45:26.772176 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 376:540, ack 1, win 501, length 164
23:45:26.772305 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 540:576, ack 1, win 501, length 36
23:45:26.772333 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 576:716, ack 1, win 501, length 140
23:45:26.772394 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 576, win 8193, length 0
23:45:26.772417 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 716:752, ack 1, win 501, length 36
23:45:26.772466 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 752:876, ack 1, win 501, length 124
23:45:26.772509 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 752, win 8192, length 0
23:45:26.772581 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 876:1052, ack 1, win 501, length 176
23:45:26.772669 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 1052, win 8191, length 0
23:45:26.772674 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1052:1088, ack 1, win 501, length 36
23:45:26.772699 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1088:1228, ack 1, win 501, length 140
23:45:26.772832 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 1228, win 8190, length 0
23:45:26.772848 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1228:1264, ack 1, win 501, length 36
23:45:26.772957 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1264:1388, ack 1, win 501, length 124
23:45:26.773036 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 1388, win 8189, length 0
23:45:26.773041 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1388:1424, ack 1, win 501, length 36
23:45:26.797688 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1424:1564, ack 1, win 501, length 140
23:45:26.797780 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1564:1740, ack 1, win 501, length 176
23:45:26.797883 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 1564, win 8195, length 0
23:45:26.797895 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1740:1916, ack 1, win 501, length 176
23:45:26.798020 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 1916, win 8194, length 0
23:45:26.798025 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 1916:2076, ack 1, win 501, length 160
23:45:26.798092 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 2076:2252, ack 1, win 501, length 176
23:45:26.798231 IP 10.0.0.170.50399 > bored.ssh: Flags [.], ack 2252, win 8192, length 0
23:45:26.798265 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 2252:2288, ack 1, win 501, length 36
23:45:26.798311 IP bored.ssh > 10.0.0.170.50399: Flags [P.], seq 2288:2428, ack 1, win 501, length 140
```

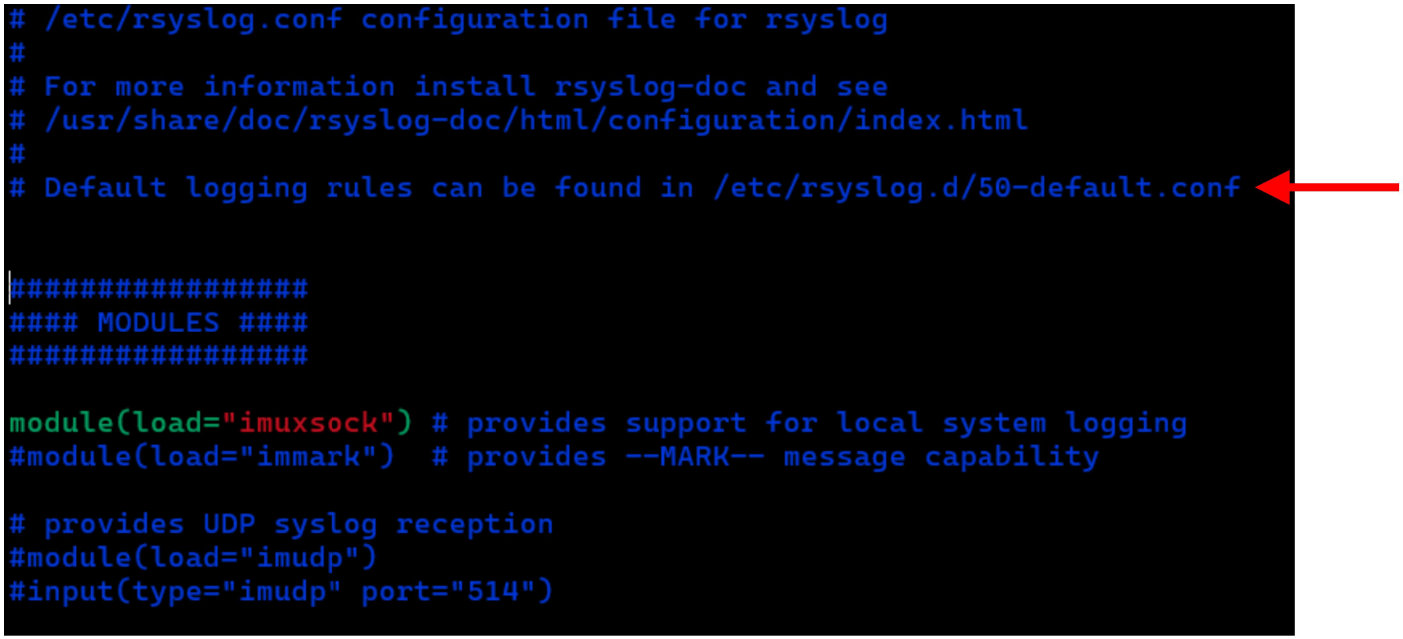
Ok, for the last capture we need to understand there are rules that tell the system what to grab, where to send it, should it alert users, and more. Lets take a look at rsyslog.conf which can be found in /etc.
I’m going to use VIM to review and edit.

```
chris@bored:/etc$ vim rsyslog.conf
```

```
# /etc/rsyslog.conf configuration file for rsyslog
#
# For more information install rsyslog-doc and see
# /usr/share/doc/rsyslog-doc/html/configuration/index.html
#
# Default logging rules can be found in /etc/rsyslog.d/50-default.conf


#################
#### MODULES ####
#################

module(load="imuxsock") # provides support for local system logging
#module(load="immark")  # provides --MARK-- message capability

# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")
```

The goal here is to collect UDP traffic over any interface. So we want to start with a rule to forward UDP traffic and add it. “*.* @10.0.0.161:514”

Though will rsyslog.conf work? It does tell us default rules are located elsewhere. Lets go look.



Ok, for the last capture we need to understand there are rules that tell the system what to grab, where to send it, should it alert users, and more. Lets take a look at rsyslog

```
chris@bored:/etc/rsyslog.d$ sudo vim 50-default.conf
```

```
chris@bored:/etc/rsyslog.d$ sudo systemctl restart rsyslog.service
chris@bored:/etc/rsyslog.d$ sudo tcpdump -i any udp port 514
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
```

```
C:\Users\Chris>ssh chris@10.0.0.161
chris@10.0.0.161's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-101-generic x86_64)
```

```
chris@bored:/etc$ cd rsyslog.d/
chris@bored:/etc/rsyslog.d$ sudo vim 50-default.conf
```

I’m going add the rule to the file and see if it works.

```
#  Default rules for rsyslog.
#
#                       For more information see rsyslog.conf(5) and /etc/rsyslog.conf

#
# First some standard log files.  Log by facility.
#
*.* @10.0.0.161:514
auth,authpriv.*                 /var/log/auth.log
*.*;auth,authpriv.none          -/var/log/syslog
#cron.*                         /var/log/cron.log
#daemon.*                       -/var/log/daemon.log
kern.*                          -/var/log/kern.log
#lpr.*                          -/var/log/lpr.log
mail.*                          -/var/log/mail.log
#user.*                         -/var/log/user.log

#
# Logging for the mail system.  Split it up so that
# it is easy to write scripts to parse these files.
#
#mail.info                      -/var/log/mail.info
#mail.warn                      -/var/log/mail.warn
mail.err                        /var/log/mail.err
```

We need to restart the service and then before we capture to a file lets verify it works.

```
chris@bored:/etc/rsyslog.d$ sudo systemctl restart rsyslog.service
chris@bored:/etc/rsyslog.d$ sudo tcpdump -i any udp port 514
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
```

Once listening, I am going to open another command prompt and ssh in. If it worked we should see logs start to populate..

```
C:\Users\Chris>ssh chris@10.0.0.161
chris@10.0.0.161's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-101-generic x86_64)
```

```
16:38:11.926090 lo    In  IP bored.33970 > bored.syslog: SYSLOG auth.info, length: 98
16:38:11.931496 lo    In  IP bored.33970 > bored.syslog: SYSLOG authpriv.info, length: 113
16:38:11.939172 lo    In  IP bored.33970 > bored.syslog: SYSLOG auth.info, length: 77
16:38:11.941373 lo    In  IP bored.33970 > bored.syslog: SYSLOG daemon.info, length: 72
16:38:22.908879 lo    In  IP bored.33970 > bored.syslog: SYSLOG auth.info, length: 110
16:38:22.909428 lo    In  IP bored.33970 > bored.syslog: SYSLOG auth.info, length: 89
16:38:22.909945 lo    In  IP bored.33970 > bored.syslog: SYSLOG authpriv.info, length: 92
16:38:22.912850 lo    In  IP bored.33970 > bored.syslog: SYSLOG auth.info, length: 101
16:38:22.913432 lo    In  IP bored.33970 > bored.syslog: SYSLOG daemon.info, length: 82
16:38:22.914611 lo    In  IP bored.33970 > bored.syslog: SYSLOG auth.info, length: 67
^C
10 packets captured
23 packets received by filter
0 packets dropped by kernel
```

Success, so now lets go ahead and capture again and this time write them to a file.

```
chris@bored:/etc$ sudo tcpdump -i any udp port 514 -w syslog_traffic.pcap
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 byt
es
^C4 packets captured
11 packets received by filter
0 packets dropped by kernel
```

Great job! There is a lot to unpack here and we have only scratched the surface.

Keep using and testing to help retain what you have learned.