Lab 08 – Deploying a Honeypot

Name: Olivia Flores
Course: IS-3033-ON2
Date: 4/23/2024

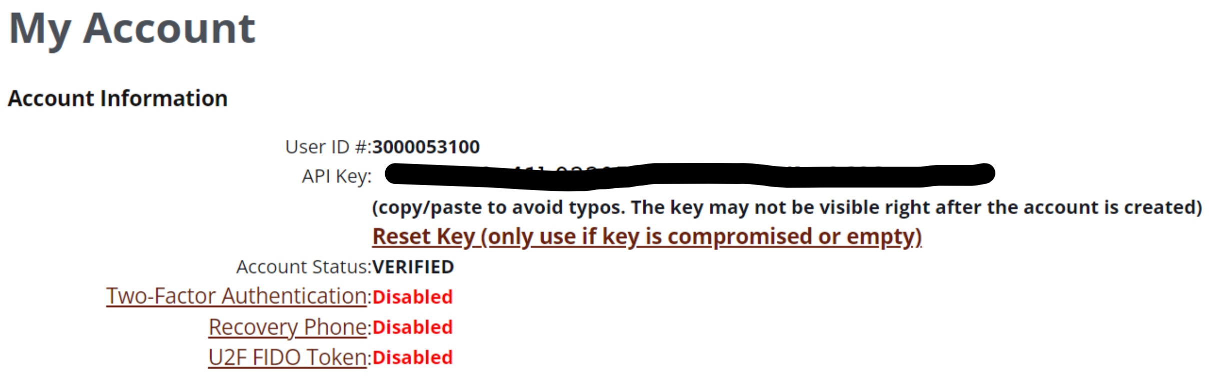
INTRODUCTION

In this lab, I will continue building on what was accomplished from Lab 07, which included being able to set up a cloud server to then SSH into that server/instance. In this lab, I will be walking you through the steps I took to set up a honeypot in a cloud environment. Some things I wanted to make sure before I began is to set up my account with the SANS Internet Storm Center (ISC), was make sure I was able to log in to my Honeypot-Tester AWS account (where I will be launching my instance again), and lastly, just have my PowerShell CLI ready to go from my host machine. I am going into this lab with zero experience in deploying honeypots, but I am excited to learn.

In Matthew O'Brien's Medium blog post, presented in the Lab 08 instructions, he introduces the honeypot I will warming up to in further steps of this lab. This honeypot is known as DShield and is known as a type of low interaction honeypot created to collect data and send it back to ISC. Furthermore, a low interaction honeypot is a type of honeypot that does not give the attacker much of an opportunity to interact with the service or application that is being stimulated [1]. With that being said, I was excited to start the lab, as it will allow me to warm up to the task of setting up a honeypot with low interaction (much safer for beginners). Some things I would like to point out from this research is that as an owner of the honeypot being deployed here, I have an advantage using DShield because I am able to receive information such as SSH and Telnet usernames and passwords via Cowrie, HTTP honeypot collecting full HTTP requests and firewall logs that will be collected from the honeypot. This information in turn, is helpful in gathering what the latest attacker trends are.

I quickly visited the SANS Internet Storm Center page to register my account using this link: https://isc.sans.edu/register.html.

From the same page I also clicked on the link to join the Slack workspace for DShield users.



My Account

Account Information

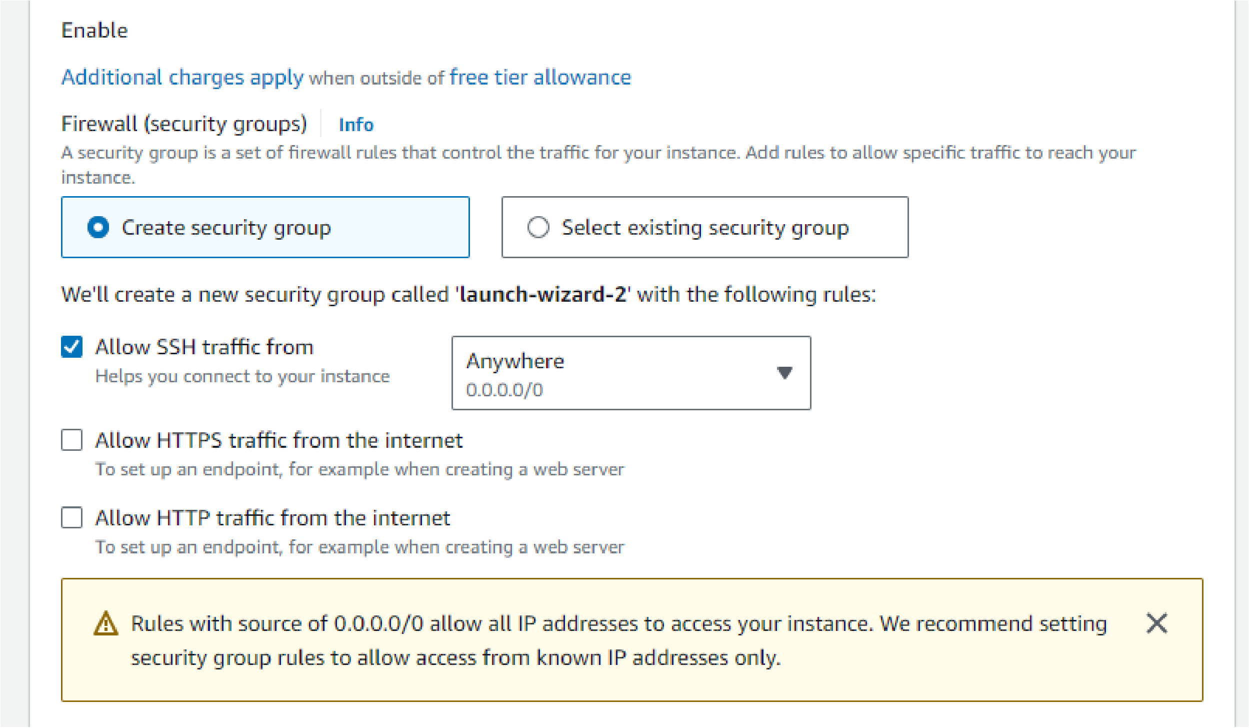
User ID #:3000053100
API Key: [redacted]
(copy/paste to avoid typos. The key may not be visible right after the account is created)
Reset Key (only use if key is compromised or empty)
Account Status:VERIFIED
Two-Factor Authentication:Disabled
Recovery Phone:Disabled
U2F FIDO Token:Disabled

Figure 1: Noting Account information from the SANS ISC home page.

BREAKPOINT 1

Step 01: SSH into an AWS Instance

As demonstrated in the previous lab, I will first be logging into my Honeypot-Tester account through AWS then launching the instance. This will function as the honeypot for this lab, so it is important to refer back to lab 07 demonstrations if needed. Here, I will leave the SSH security group setting as is shown in the figure below, noting the warning in the yellow box allowing all IP addresses to access this instance.



Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

We'll create a new security group called '**launch-wizard-2**' with the following rules:

☑ Allow SSH traffic from — Anywhere 0.0.0.0/0

Helps you connect to your instance

☐ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

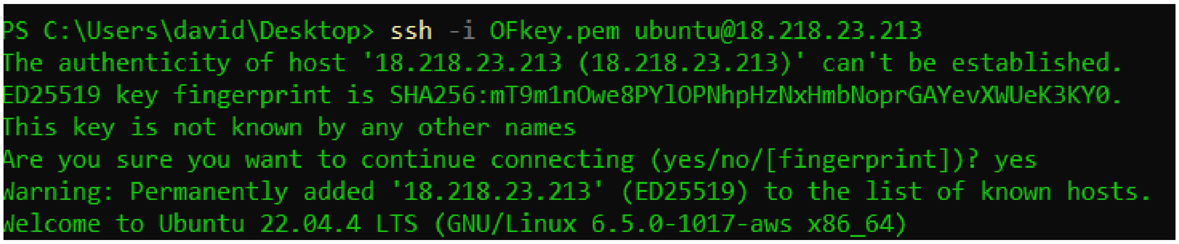
☐ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Figure 2: Launching an instance of an AWS machine.

Next, I want to test my instance by SSH'ing into it from my host machine. I used the PowerShell CLI through the "Run as Administrator" function. One thing I did not point out in the previous lab is that I am using my own laptop, but never reset the settings to reflect my own information as olivia instead of my partner who was the previous owner. Therefore, in this lab, the CLI will still reflect my partner's name, as "david." Down below you can see I successfully ssh'ed into the instance.



```
PS C:\Users\david\Desktop> ssh -i OFkey.pem ubuntu@18.218.23.213
The authenticity of host '18.218.23.213 (18.218.23.213)' can't be established.
ED25519 key fingerprint is SHA256:mT9m1nOwe8PY1OPNhpHzNxHmbNoprGAYevXWUeK3KY0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '18.218.23.213' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.5.0-1017-aws x86_64)
```

Figure 3: Successful SSH into ubuntu instance.

I did not run into any issues in this step of the lab as it was more of setting up the environment to then complete the rest of the lab. I am certain that I will have to repeat this step a couple of times in the case that I have to experiment with deploying a honeypot successfully, therefore there might be a change in the IP Address that I will be showing throughout this lab.

BREAKPOINT 2

Step 02: Set Up the Python Environment

First thing is first, when already SSH'ed into my ubuntu machine I am going to want to update the OS using the following command:

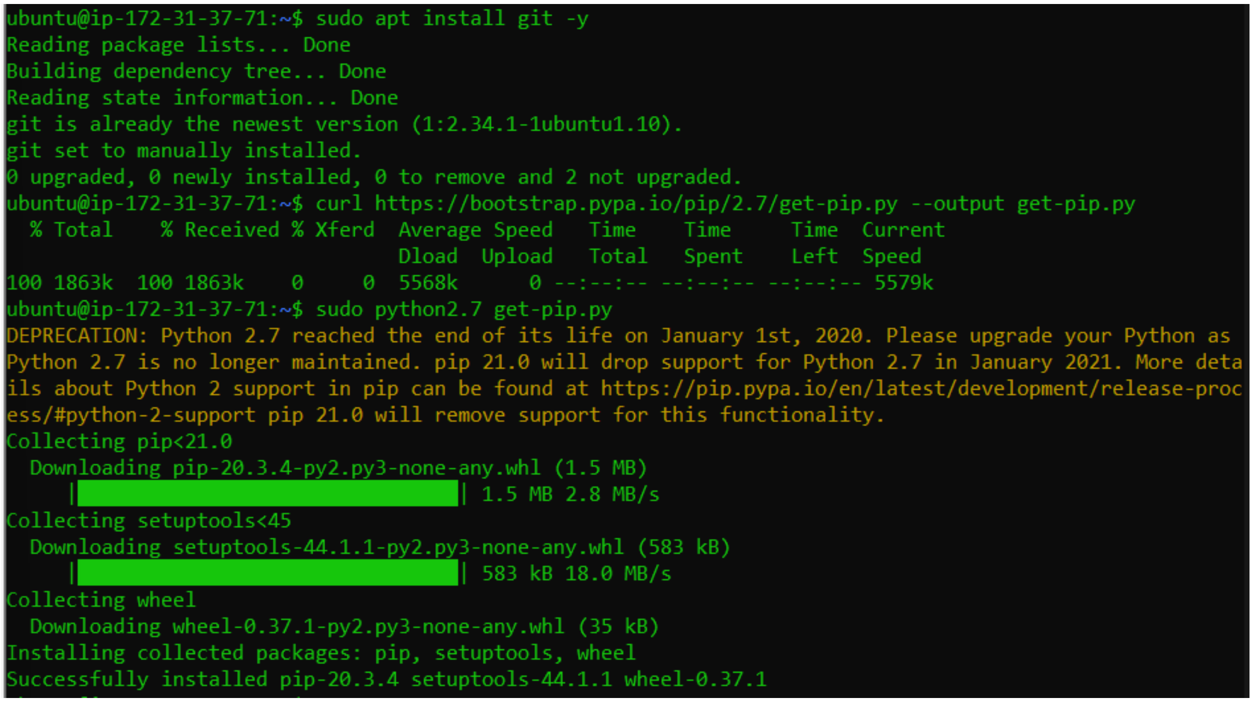
sudo apt update && sudo apt full-upgrade -y

After I ran that command, there was an option to reboot the system shortly after, so I pressed "ok." I noticed when I did the reboot, it automatically closed my SSH session with the ubuntu instance, so I had to go back and SSH in again using the command displayed at the top of figure 3.

Now I want to make sure the honeypot (built on Cowrie) has a Python environment to be able to run so we will continue with installing the necessary Python libraries in the home directory of the ubuntu server. Please see the following commands that I used one after another:

```
cd ~
sudo apt install python3-pip -y
sudo apt install python2.7 -y
sudo apt install git -y
curl https://bootstrap.pypa.io/pip/2.7/get-pip.py --output get-pip.py
```

Lastly, I installed the Python interpreter, version 2.7 to then be able to execute the **get-pip.py** script using: **sudo python2.7 get-pip.py**

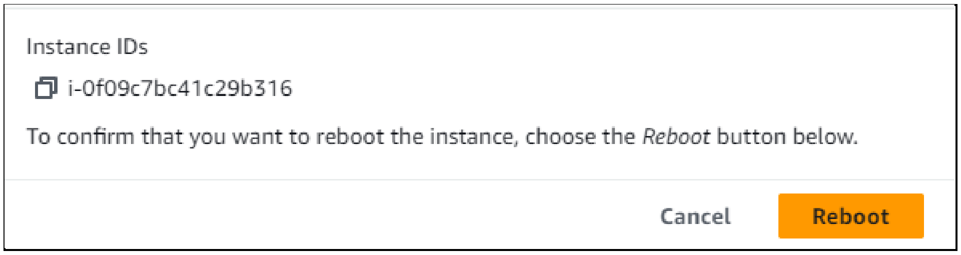


```
ubuntu@ip-172-31-37-71:~$ sudo apt install git -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.10).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
ubuntu@ip-172-31-37-71:~$ curl https://bootstrap.pypa.io/pip/2.7/get-pip.py --output get-pip.py
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 1863k  100 1863k    0     0  5568k      0 --:--:-- --:--:-- --:--:-- 5579k
ubuntu@ip-172-31-37-71:~$ sudo python2.7 get-pip.py
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Please upgrade your Python as
Python 2.7 is no longer maintained. pip 21.0 will drop support for Python 2.7 in January 2021. More deta
ils about Python 2 support in pip can be found at https://pip.pypa.io/en/latest/development/release-proc
ess/#python-2-support pip 21.0 will remove support for this functionality.
Collecting pip<21.0
  Downloading pip-20.3.4-py2.py3-none-any.whl (1.5 MB)
     |████████████████████████████████| 1.5 MB 2.8 MB/s
Collecting setuptools<45
  Downloading setuptools-44.1.1-py2.py3-none-any.whl (583 kB)
     |████████████████████████████████| 583 kB 18.0 MB/s
Collecting wheel
  Downloading wheel-0.37.1-py2.py3-none-any.whl (35 kB)
Installing collected packages: pip, setuptools, wheel
Successfully installed pip-20.3.4 setuptools-44.1.1 wheel-0.37.1
```

Figure 4: Successful install of Python 2.7 interpreter.

In the listed commands above, we are changing to our home directory and then installing pip for Python3. We also installed Python2.7 and git. Finally, to install pip2, we must grab a script from pypa.io and execute it using the curl command in the figure shown above.

Next, I went back to the AWS console, to reboot the instance. The Connection to this IP address then closed, so I had to go back and SSH in again. I encountered an issue where I was trying to connect, but the connection would time out so I would simply attempt to SSH in for a second time and it worked. After this step has been complete, I am ready to install the DShield sensor.



Instance IDs

i-0f09c7bc41c29b316

To confirm that you want to reboot the instance, choose the *Reboot* button below.

Figure 5: Rebooting AWS instance from the console.

BREAKPOINT 3

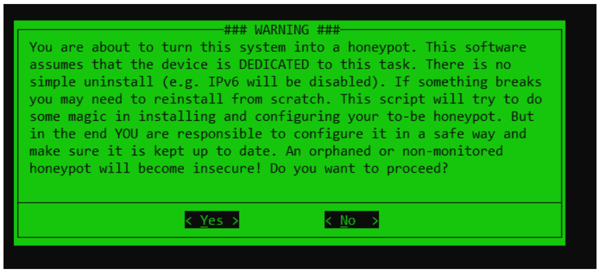
Step 03: Install the DShield Sensor

Now to start the installation for DShield, I used the instructions in our lab and from the Matthew O'Briend's blog post. These instructions have been gathered from the DShield installation Guide PDF as well but have been modified to include a bit more details and demonstration. This process is fairly easy, as I just have to run the commands as follows, but it can start getting a bit tricky in the further steps, dealing with "Additional IPs" so stay tuned for the process.

1. First, I will run the following commands from my home directory one after another:

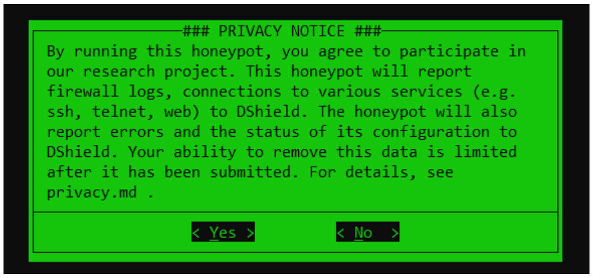
```
mkdir install
cd install
git clone https://github.com/DShield-ISC/dshield.git
cd dshield/bin
sudo ./install.sh
```

2. A GUI installer will begin to walk us through this process. I was prompted with a message warning about the risks of creating a honeypot and a privacy notice. I have selected "Yes" for both. To toggle between the options, I can just use the arrow keys and to pick an answer, I pressed **Enter**.



```
### WARNING ###
You are about to turn this system into a honeypot. This software
assumes that the device is DEDICATED to this task. There is no
simple uninstall (e.g. IPv6 will be disabled). If something breaks
you may need to reinstall from scratch. This script will try to do
some magic in installing and configuring your to-be honeypot. But
in the end YOU are responsible to configure it in a safe way and
make sure it is kept up to date. An orphaned or non-monitored
honeypot will become insecure! Do you want to proceed?

        < Yes >            < No  >
```

Figure 6: Warning about the risks of creating a honeypot.

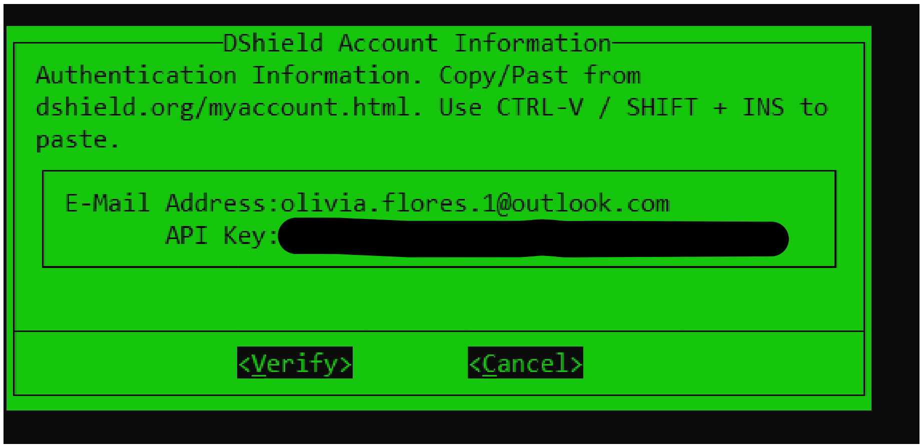


```
### PRIVACY NOTICE ###
By running this honeypot, you agree to participate in
our research project. This honeypot will report
firewall logs, connections to various services (e.g.
ssh, telnet, web) to DShield. The honeypot will also
report errors and the status of its configuration to
DShield. Your ability to remove this data is limited
after it has been submitted. For details, see
privacy.md .

        < Yes >            < No  >
```

Figure 7: Privacy notice for creating a honeypot.

3. Next, I was asked to choose between **manual** or **automatic** updates. I chose **automatic**.

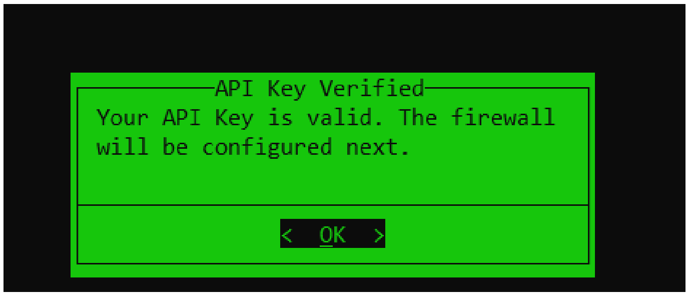
4. For the DShield Account Information pop up, we will be needing to pull up the information we saved in the start of this lab. I input the email I used to set up my **DShield** account and the **API** key which is available under the "My Account" page on my **DShield** account. I copied this key directly from my account to avoid errors. To toggle from the email address to API Key, it is important to use the *down arrow.* I accidentally pressed tab and was not able to go back and input my API Key, having to redo the process. Next, I pressed Enter to "Verify".



```
DShield Account Information
Authentication Information. Copy/Past from
dshield.org/myaccount.html. Use CTRL-V / SHIFT + INS to
paste.

E-Mail Address:olivia.flores.1@outlook.com
       API Key:[redacted]
```

Figure 8: Authenticating to DShield.

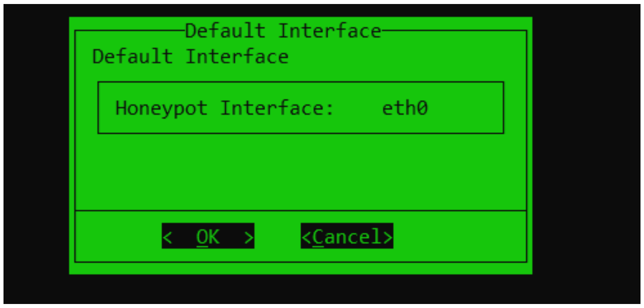
5. Since everything checked out, I was able to get the successful message displayed below, for "API Key Verified."



```
API Key Verified
Your API Key is valid. The firewall
will be configured next.
```

Figure 9: Moving onto the firewall.

6. I selected the interface to be eth0 which is already input by default.

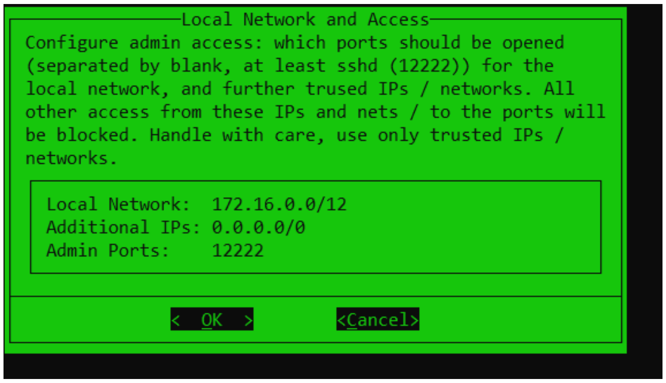


```
Default Interface
Default Interface

Honeypot Interface:    eth0
```

Figure 10: Setting the default network interface.

7. After this, I will configure the “Local Network and Access”. These are the ports and IPs that have administrative access to the honeypot. The default admin port is **12222**, which I will leave the same. For the “Local Network”, it pre-filled the **CIDR range** of the local network that the EC2 instance resides in.

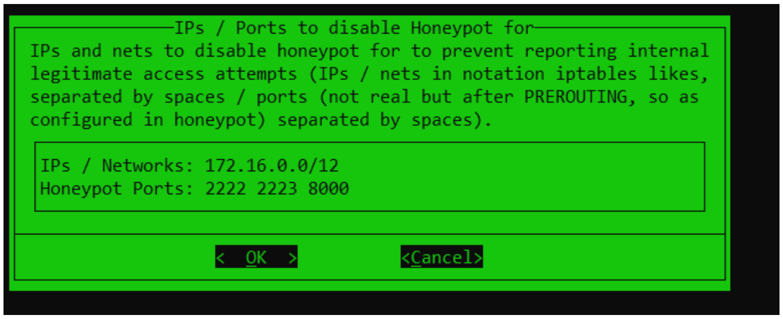
8. For the **Additional IPs** section, it seems that I will also need to allow my host here to access port 12222, even if this is not explicitly stated. When I got to this section, I first went to disable my VPN for the purposes of testing the honeypot. I was not sure if my address was static or dynamic, but to test the parameter with the least restrictions, I simply tried with trusting any IP address: **0.0.0.0/0**. Next, I pressed Ok.



```
Local Network and Access
Configure admin access: which ports should be opened
(separated by blank, at least sshd (12222)) for the
local network, and further trused IPs / networks. All
other access from these IPs and nets / to the ports will
be blocked. Handle with care, use only trusted IPs /
networks.

  Local Network:  172.16.0.0/12
  Additional IPs: 0.0.0.0/0
  Admin Ports:    12222
```

Figure 11: Allowing my host to access the honeypot through port 12222.

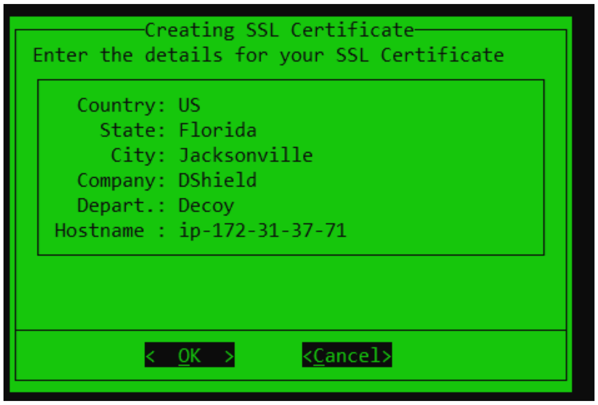
9. We will then be asked if there are any IPs that the firewall *should not* log or redirect to honeypot ports. The CIDR range for the local network should be used here. A host static IP address could also be placed here. I clicked “Ok” for this and the confirmation message afterwards. Similarly, we will be asked for ports and IPs to for which to disable the honeypot to prevent logging of legitimate access attempts. The CIDR range for the local network should suffice and we should not have to change any of the honeypot ports. I clicked “Ok” for this and the confirmation message afterwards.



```
IPs / Ports to disable Honeypot for
IPs and nets to disable honeypot for to prevent reporting internal
legitimate access attempts (IPs / nets in notation iptables likes,
separated by spaces / ports (not real but after PREROUTING, so as
configured in honeypot) separated by spaces).

IPs / Networks: 172.16.0.0/12
Honeypot Ports: 2222 2223 8000
```

Figure 12: IPs listed that the firewall should not log.

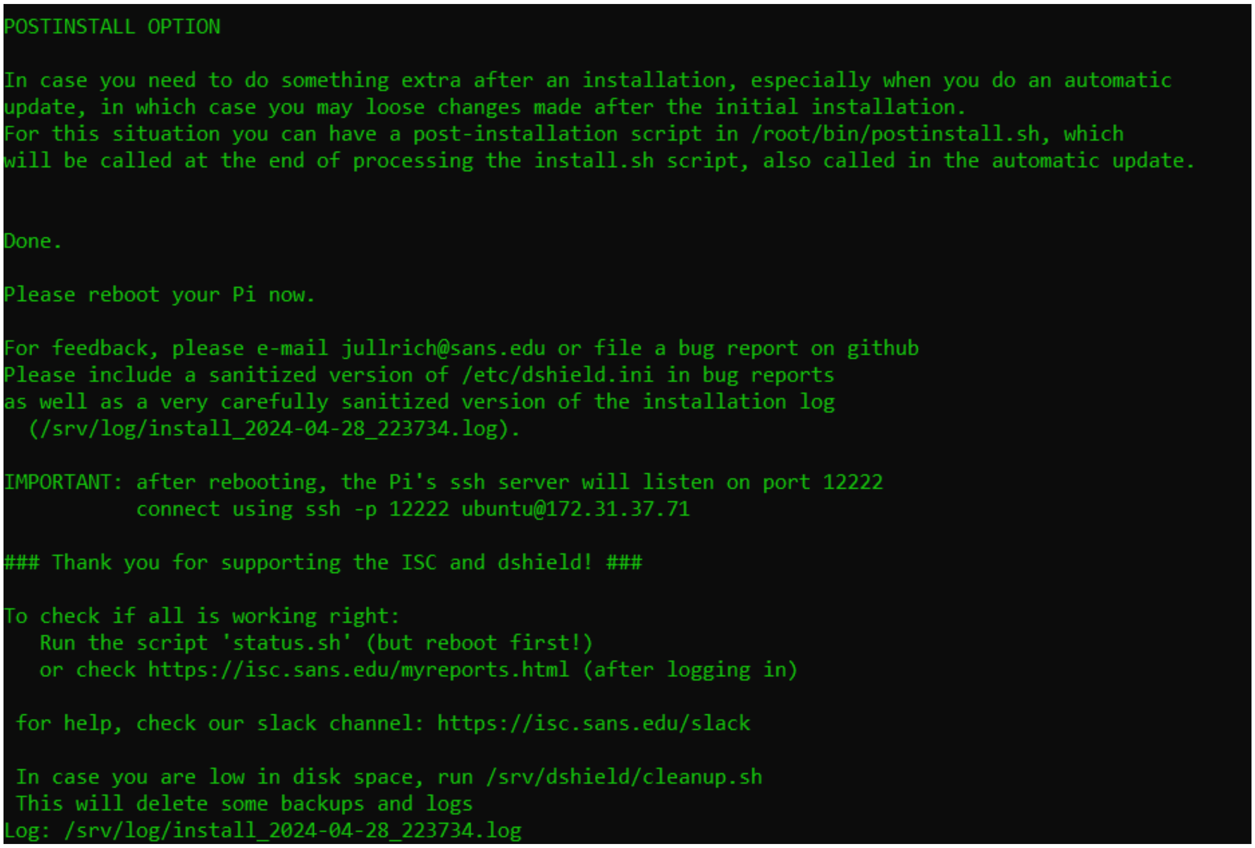
10. Next, I must create an SSL certificate. I went with the defaults here and clicked “Ok.” Shortly after, it asked me if I wanted it to create a CA for it to sign the certificate. I chose “Yes”.



```
Creating SSL Certificate
Enter the details for your SSL Certificate

    Country: US
      State: Florida
       City: Jacksonville
    Company: DShield
    Depart.: Decoy
  Hostname : ip-172-31-37-71
```

Figure 13: Creating SSL Certificate.

11. Finally, once the installation was complete, I did a reboot to my EC2 instance from the console. I also wanted to make note of the following information for future reference and noting the "Important" message.



```
POSTINSTALL OPTION

In case you need to do something extra after an installation, especially when you do an automatic
update, in which case you may loose changes made after the initial installation.
For this situation you can have a post-installation script in /root/bin/postinstall.sh, which
will be called at the end of processing the install.sh script, also called in the automatic update.


Done.

Please reboot your Pi now.

For feedback, please e-mail jullrich@sans.edu or file a bug report on github
Please include a sanitized version of /etc/dshield.ini in bug reports
as well as a very carefully sanitized version of the installation log
  (/srv/log/install_2024-04-28_223734.log).

IMPORTANT: after rebooting, the Pi's ssh server will listen on port 12222
           connect using ssh -p 12222 ubuntu@172.31.37.71

### Thank you for supporting the ISC and dshield! ###

To check if all is working right:
   Run the script 'status.sh' (but reboot first!)
   or check https://isc.sans.edu/myreports.html (after logging in)

 for help, check our slack channel: https://isc.sans.edu/slack

 In case you are low in disk space, run /srv/dshield/cleanup.sh
 This will delete some backups and logs
Log: /srv/log/install_2024-04-28_223734.log
```

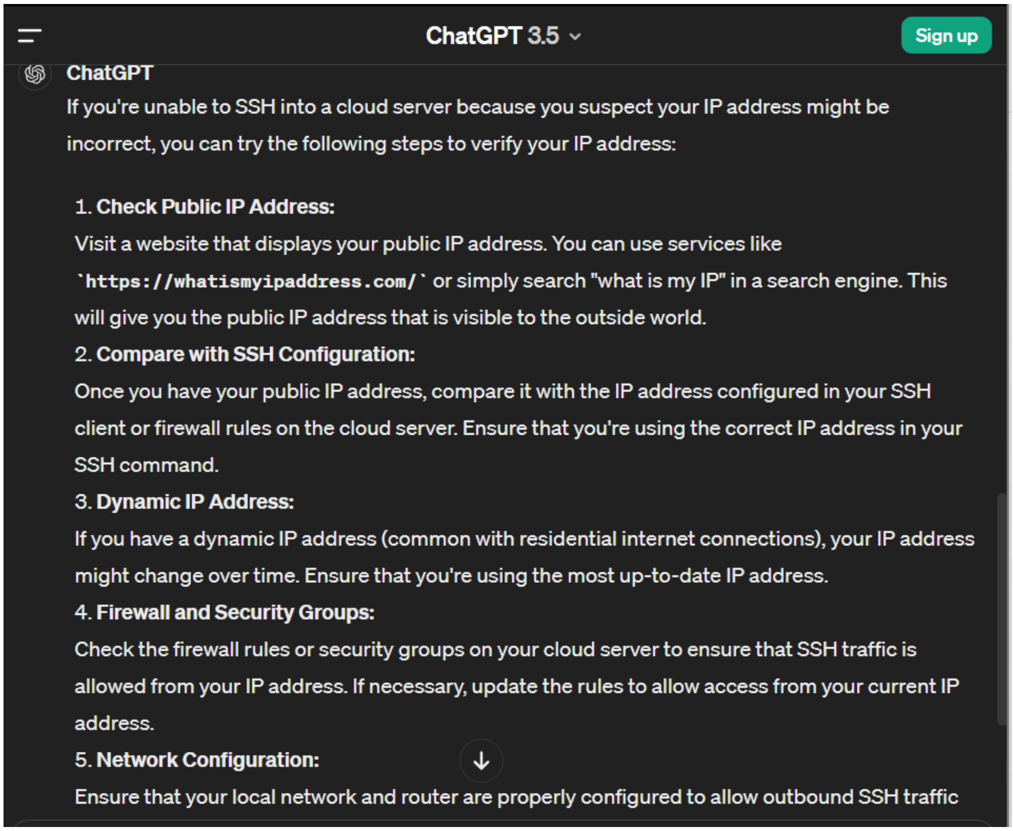
Figure 14: Noting post installation tips.

BREAKPOINT 4

Step 04: Test Your Access

Now I want to test my access by SSHing into the honeypot through port 12222. I will take you through with me on how I had to experiment with the **Security Group** rules, going from least restrictive to most restrictive. From this point forward, it did take me lots of experimenting, therefore, I will have changes in my IP addresses, as I simply had to create multiple instances to change the Security Group settings at times. I also could find hints along the way pointed out from classmates and the professor through our Slack module 8 channel and through re-reading instructions multiple times. At one point, I simply chose the wrong server to launch my instance on and realized I could not install python2.7 due to the version I was on, so I had to restart. Another time, I was simply running into hiccups when I was installing the DShield sensor because of the "Additional IPs" setting I was testing.

A safe measure is to create a **Security Group** that only allows SSH traffic on port 12222 from my IP address. Here I went to the CLI on command prompt to see my IP address using the **ipconfig** command, so when I went to the Security Group settings, I input the IP address 192.168.1.9/24, but when I tried SSHing into the machine through port 12222, I continued getting no results, so I knew I was probably not using the correct address. It was quite a simple issue, but I was unsure so I quickly went to ask generative AI ChatGPT a question, "How do i find my ip address?" and they responded with a few tips to use my Command Line prompt and double checking my settings, which I had already done, but it still was not the answer I was looking for. I asked another follow up question, "what if i am trying to ssh through port 12222 into a cloud server from my ip address, but i am not able to because the ip address is incorrect? how can i find the correct ip address?" Some of the responses I got are shown below [2]. This was helpful in learning that my IP address could be a dynamic IP address that gets masked through my VPN and clearly I was using the incorrect IP address in my security group settings. When I conducted a search through the google engine, I was able to see the IP address that is visible to the outside world.



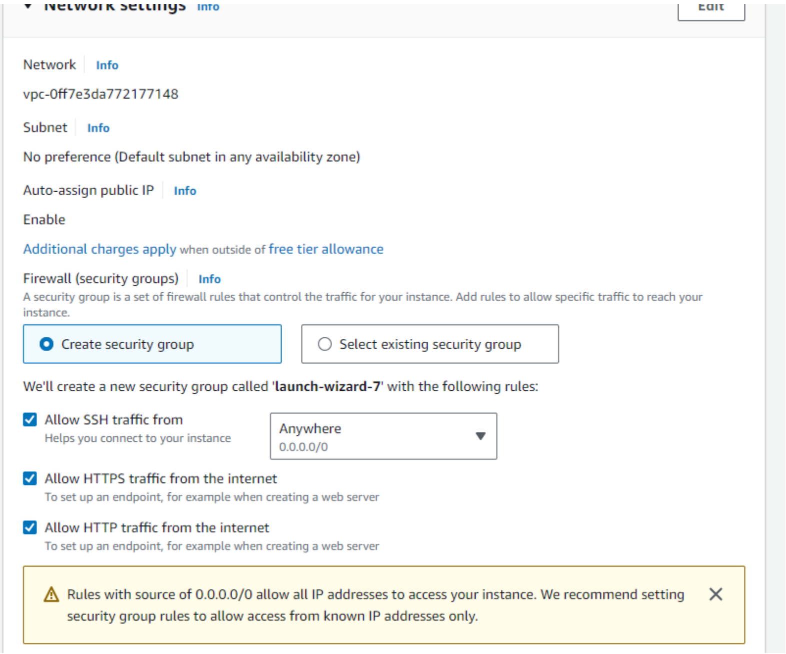
ChatGPT

If you're unable to SSH into a cloud server because you suspect your IP address might be incorrect, you can try the following steps to verify your IP address:

1. **Check Public IP Address:**
Visit a website that displays your public IP address. You can use services like `https://whatismyipaddress.com/` or simply search "what is my IP" in a search engine. This will give you the public IP address that is visible to the outside world.
2. **Compare with SSH Configuration:**
Once you have your public IP address, compare it with the IP address configured in your SSH client or firewall rules on the cloud server. Ensure that you're using the correct IP address in your SSH command.
3. **Dynamic IP Address:**
If you have a dynamic IP address (common with residential internet connections), your IP address might change over time. Ensure that you're using the most up-to-date IP address.
4. **Firewall and Security Groups:**
Check the firewall rules or security groups on your cloud server to ensure that SSH traffic is allowed from your IP address. If necessary, update the rules to allow access from your current IP address.
5. **Network Configuration:**
Ensure that your local network and router are properly configured to allow outbound SSH traffic

Figure 15: ChatGPT response to ip address question.

After noticing this error, I also noticed other classmates running into an issue where they could not SSH into their honeypot due to the restrictions set up on the instance and I continued noticing more feedback from our professor which included tips along the lines of widening permissions first. This was an issue for me when initially launching the instance so all the previous steps would have to be redone. I got a bit discouraged, but I also realized that these steps could be easily overlooked, and I was not the only one running into hiccups. I wanted to ensure I was following all the correct steps, so I began from the top, creating another instance, but this time allowing SSH traffic from anywhere and allowing HTTPS, HTTP traffic from the internet as displayed in the Network settings below. This was another setting I overlooked as I was also aware of the potential the honeypot has in collecting full HTTP requests and firewall logs.



Network Info
vpc-0ff7e3da772177148

Subnet Info
No preference (Default subnet in any availability zone)

Auto-assign public IP Info
Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

We'll create a new security group called '**launch-wizard-7**' with the following rules:

☑ Allow SSH traffic from — Anywhere 0.0.0.0/0
Helps you connect to your instance

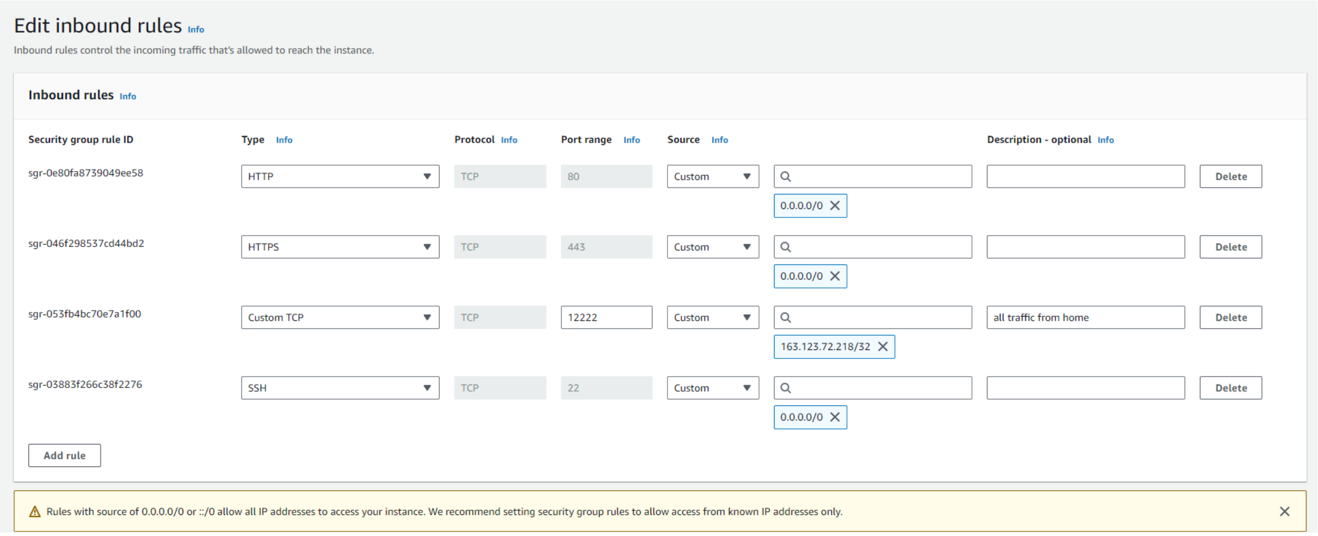
☑ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

☑ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Figure 16: Configuring Network settings and creating security group.

Next, I went through all the steps again to install DShield on the new instance, using the "Additional IPs" as **0.0.0.0/0** for the purposes of the honeypot. I also went and added another security group setting that allowed SSH traffic on port 12222 from my public IP address.



Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

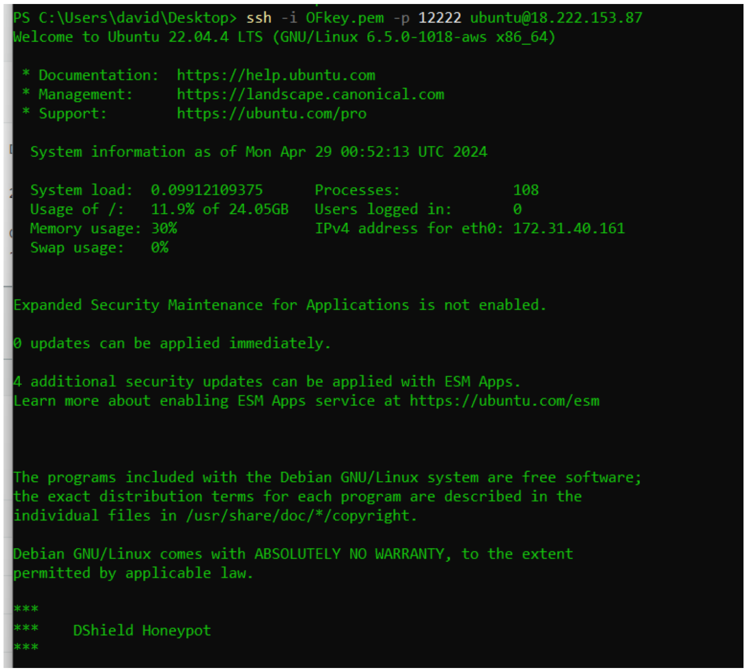
Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-0e80fa8739049ee58	HTTP	TCP	80	Custom	0.0.0.0/0	
sgr-046f298537cd44bd2	HTTPS	TCP	443	Custom	0.0.0.0/0	
sgr-053fb4bc70e7a1f00	Custom TCP	TCP	12222	Custom	163.123.72.218/32	all traffic from home
sgr-03883f266c38f2276	SSH	TCP	22	Custom	0.0.0.0/0	

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

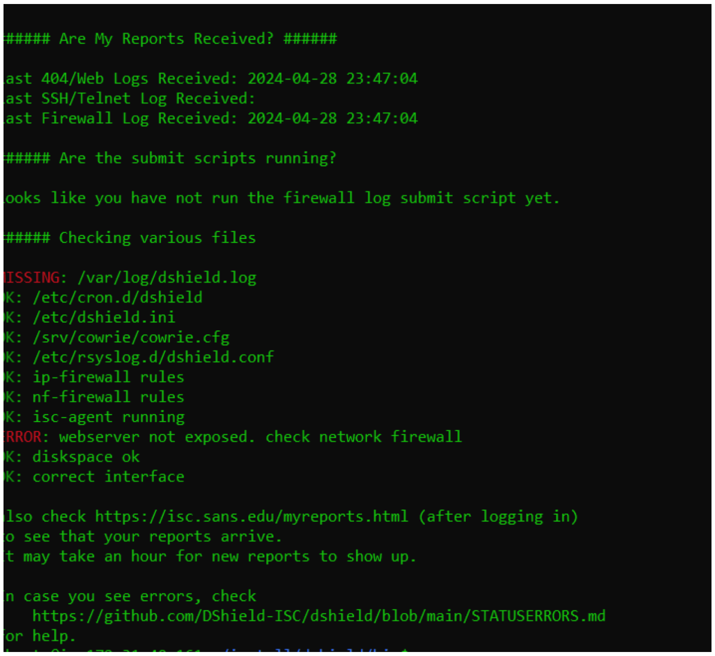
Figure 17: Allowing SSH traffic from home IP address through port 12222.

After that was configured, I was able to successfully SSH into the instance using the command **ssh -i OFkey.pem -p 12222 ubuntu@mypublicipaddress**. So now I am one step closer, but I was still running into the issue where I could not get my webserver exposed, also not being able to get the logging done as the log file was "missing." This was after I read over some tips given by some of my peers on the slack channel. From their experience I could infer that this was not an issue with my router, because the server is in the cloud, so I had to backtrack again and make some necessary changes to my settings. The output I got when SSHing into the instance is shown in the figures below.



```
PS C:\Users\david\Desktop> ssh -i OFkey.pem -p 12222 ubuntu@18.222.153.87
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.5.0-1018-aws x86_64)

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

  System information as of Mon Apr 29 00:52:13 UTC 2024

  System load:  0.09912109375     Processes:             108
  Usage of /:   11.9% of 24.05GB  Users logged in:       0
  Memory usage: 30%               IPv4 address for eth0: 172.31.40.161
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

4 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm



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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

***
***     DShield Honeypot
***
```

Figure 18: Successful SSH into instance using port 12222.

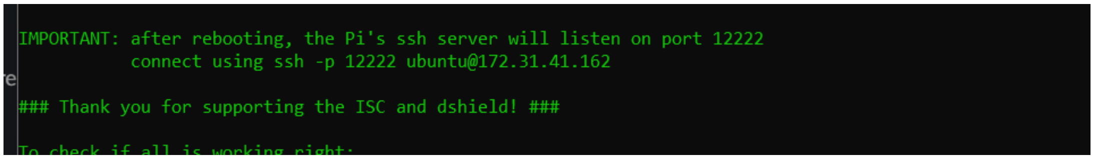


```
##### Are My Reports Received? ######

ast 404/Web Logs Received: 2024-04-28 23:47:04
ast SSH/Telnet Log Received:
ast Firewall Log Received: 2024-04-28 23:47:04

##### Are the submit scripts running?

ooks like you have not run the firewall log submit script yet.

##### Checking various files

ISSING: /var/log/dshield.log
K: /etc/cron.d/dshield
K: /etc/dshield.ini
K: /srv/cowrie/cowrie.cfg
K: /etc/rsyslog.d/dshield.conf
K: ip-firewall rules
K: nf-firewall rules
K: isc-agent running
RROR: webserver not exposed. check network firewall
K: diskspace ok
K: correct interface

lso check https://isc.sans.edu/myreports.html (after logging in)
o see that your reports arrive.
t may take an hour for new reports to show up.

n case you see errors, check
   https://github.com/DShield-ISC/dshield/blob/main/STATUSERRORS.md
or help.
```

Figure 19: Webserver not exposed, no logging done.

Now, I was not sure if I ran into the issue when I was installing DShield or if this was a simple security group setting change, I made, but I wanted to backtrack and begin a new instance again and repeat the steps. When installing DShield, I input the public IP address I was seeing from the google search, “what is my ip address?” I also had overlooked this step in the installation process, but my IP address was already on the “Additional IPs” by default. My initial thought process was that if I allow *any* IP address access, I would be able to accomplish the deploying of my honeypot regardless, but here I just left it as the default of my IP address. I pressed ok and continued to the next steps of the installation again.

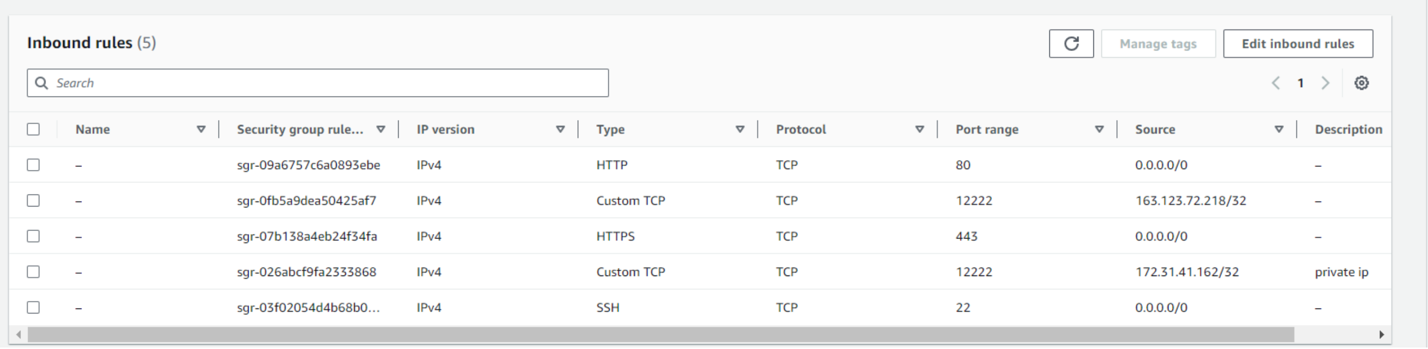
After the installation I wanted to make note of the Post install option, which I had quickly looked over the first time. In this post installation they also left an “IMPORTANT” note for me, that I should reboot and “the Pi’s ssh server will listen on port 12222 connect using **ssh -p ubuntu@privateipaddress**”



```
IMPORTANT: after rebooting, the Pi's ssh server will listen on port 12222
           connect using ssh -p 12222 ubuntu@172.31.41.162

### Thank you for supporting the ISC and dshield! ###

To check if all is working right:
```

Figure 20: Post install option on DShield.

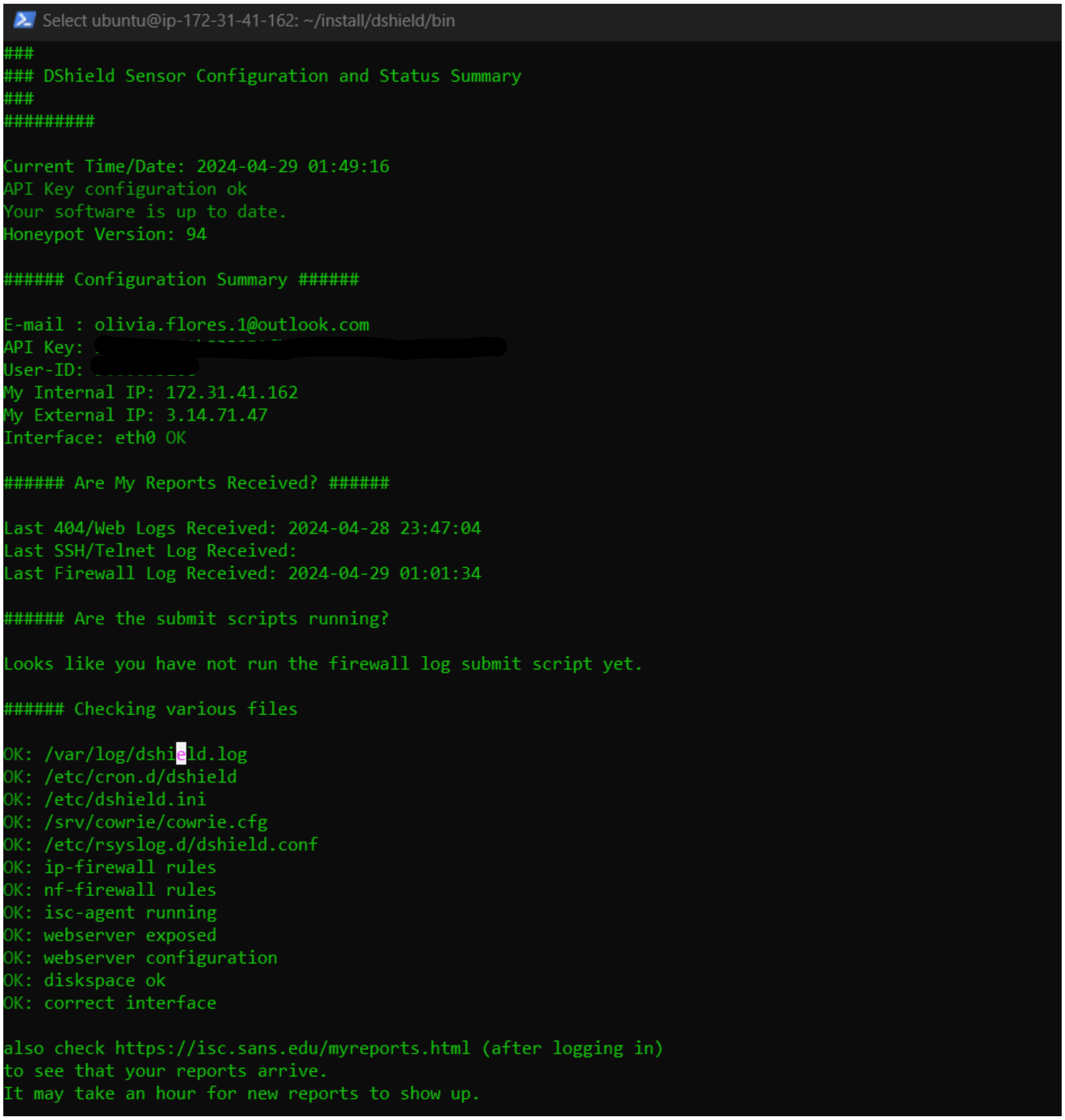
I decided to try adding another inbound rule to the instance that included my private IP address to allow traffic on port 12222. After changing the security group settings to include my private IP address, I went back and navigated to the **/home/ubuntu/install/dshield/bin** directory where I had installed DShield and tried running **status.sh**. I was able to see that there was a successful output on figure 21 and I can see that there is logging being done now.



Inbound rules (5)

☐	Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
☐	–	sgr-09a6757c6a0893ebe	IPv4	HTTP	TCP	80	0.0.0.0/0	–
☐	–	sgr-0fb5a9dea50425af7	IPv4	Custom TCP	TCP	12222	163.123.72.218/32	–
☐	–	sgr-07b138a4eb24f34fa	IPv4	HTTPS	TCP	443	0.0.0.0/0	–
☐	–	sgr-026abcf9fa2333868	IPv4	Custom TCP	TCP	12222	172.31.41.162/32	private ip
☐	–	sgr-03f02054d4b68b0...	IPv4	SSH	TCP	22	0.0.0.0/0	–

Figure 21: Adjusting inbound rules for my instance.

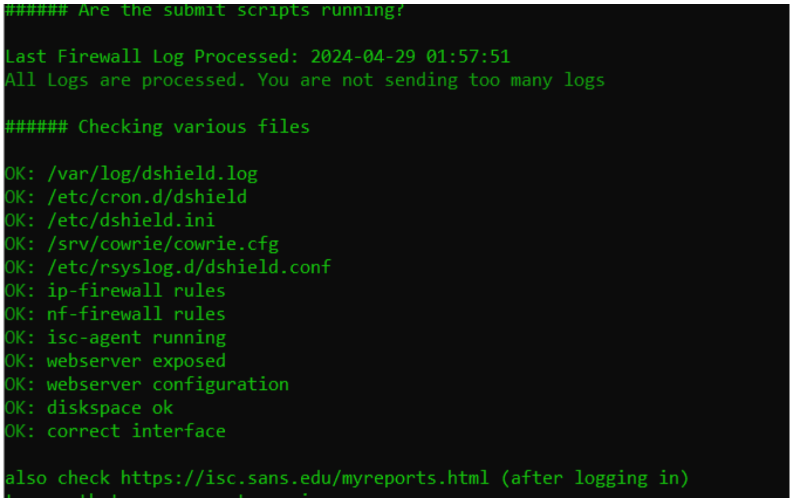


```
###
### DShield Sensor Configuration and Status Summary
###
#########

Current Time/Date: 2024-04-29 01:49:16
API Key configuration ok
Your software is up to date.
Honeypot Version: 94

###### Configuration Summary ######

E-mail : olivia.flores.1@outlook.com
API Key: [redacted]
User-ID: [redacted]
My Internal IP: 172.31.41.162
My External IP: 3.14.71.47
Interface: eth0 OK

###### Are My Reports Received? ######

Last 404/Web Logs Received: 2024-04-28 23:47:04
Last SSH/Telnet Log Received:
Last Firewall Log Received: 2024-04-29 01:01:34

###### Are the submit scripts running?

Looks like you have not run the firewall log submit script yet.

###### Checking various files

OK: /var/log/dshield.log
OK: /etc/cron.d/dshield
OK: /etc/dshield.ini
OK: /srv/cowrie/cowrie.cfg
OK: /etc/rsyslog.d/dshield.conf
OK: ip-firewall rules
OK: nf-firewall rules
OK: isc-agent running
OK: webserver exposed
OK: webserver configuration
OK: diskspace ok
OK: correct interface

also check https://isc.sans.edu/myreports.html (after logging in)
to see that your reports arrive.
It may take an hour for new reports to show up.
```

Figure 22: Successful output from DShield.

After I ran the **status.sh** shown above, I was starting to realize one thing I should have done differently was to hold off on changing the security group setting as I had already made changes to the "Additional IPs" section by simply including my home public IP address in the process when installing DShield. So, to test if this was the reason why I had a successful output, I wanted to go back to the inbound security group settings and remove my private IP address from being able to access the instance through port 12222, then repeating the steps after rebooting.

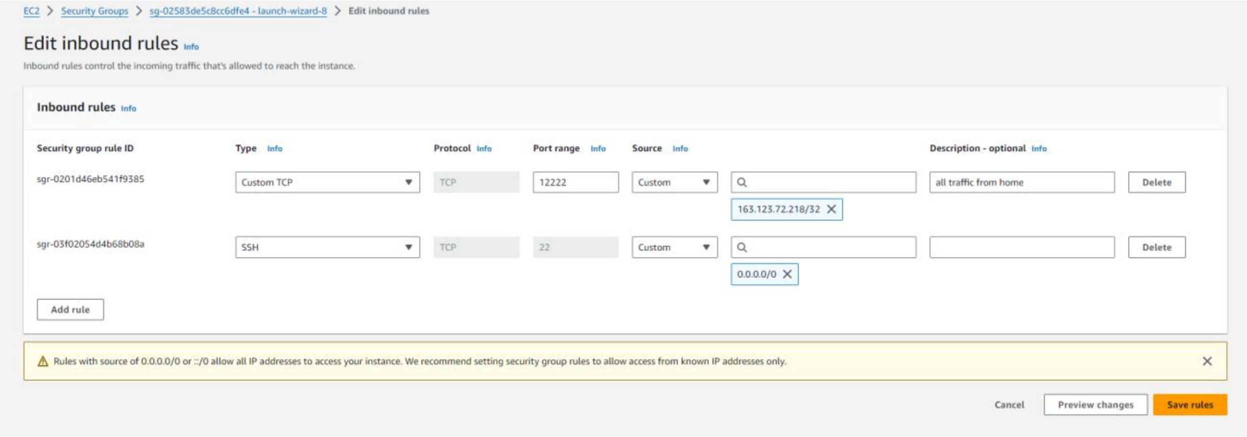
After I SSH'ed back in and ran **status.sh**, I was still able to see a successful output of the run with successful logging as shown below in the Configuration summary. So, I concluded that next time I just need to add my home IP address to the Additional IPs that will be listed as trusted IPs/networks.



```
###### Are the submit scripts running?

Last Firewall Log Processed: 2024-04-29 01:57:51
All Logs are processed. You are not sending too many logs

###### Checking various files

OK: /var/log/dshield.log
OK: /etc/cron.d/dshield
OK: /etc/dshield.ini
OK: /srv/cowrie/cowrie.cfg
OK: /etc/rsyslog.d/dshield.conf
OK: ip-firewall rules
OK: nf-firewall rules
OK: isc-agent running
OK: webserver exposed
OK: webserver configuration
OK: diskspace ok
OK: correct interface

also check https://isc.sans.edu/myreports.html (after logging in)
```

Figure 23: Successful SSH on port 12222 after removing Private IP address.

Shortly after, I changed my inbound settings again to remove allowing HTTPS and HTTP traffic from the internet, expecting this to remove the ability of the webserver being exposed which as a result, is what had happened in figure 25.



EC2 > Security Groups > sg-02583de5c8cc6dfe4 - launch-wizard-8 > Edit inbound rules

Edit inbound rules

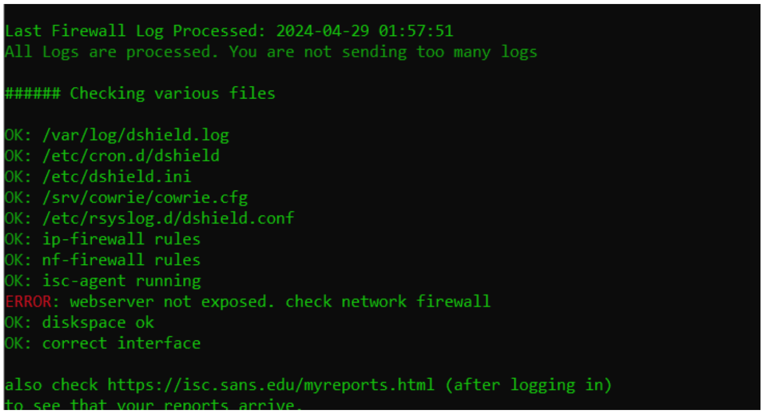
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-0201d46eb541f9385	Custom TCP	TCP	12222	Custom	163.123.72.218/32	all traffic from home
sgr-03f02054d4b68b08a	SSH	TCP	22	Custom	0.0.0.0/0	

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Figure 24: Removing setting to allow HTTPS & HTTP traffic on instance.



```
Last Firewall Log Processed: 2024-04-29 01:57:51
All Logs are processed. You are not sending too many logs

###### Checking various files

OK: /var/log/dshield.log
OK: /etc/cron.d/dshield
OK: /etc/dshield.ini
OK: /srv/cowrie/cowrie.cfg
OK: /etc/rsyslog.d/dshield.conf
OK: ip-firewall rules
OK: nf-firewall rules
OK: isc-agent running
ERROR: webserver not exposed. check network firewall
OK: diskspace ok
OK: correct interface

also check https://isc.sans.edu/myreports.html (after logging in)
to see that your reports arrive.
```

Figure 25: Webserver error when removing HTTPS & HTTP setting.

In this portion of the lab, I found myself to be very unsure of how the honeypot was supposed to work but was able to warm up with setting one up. I found that I am still needing to do more research in understanding how the ports are translated through DShield and how to verify this information once there has been a successful status.sh run through port 12222.

Breakpoint 5

Step 05: Check Your Logs!

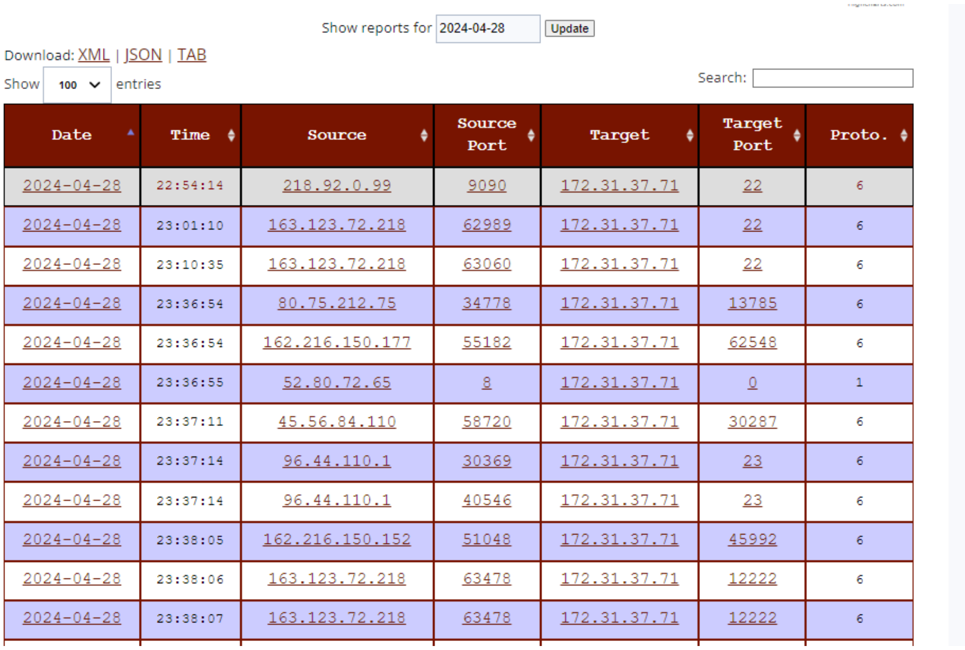
In this step of the lab, I will be visiting my DShield account to review the log reports that were created. I did have to visit back the following day because of the inconvenience of time but it allowed me to stop my instance for that day and not have to terminate it as I wanted to continue working on it.

Instance	i-06d1185a068765f77	Stopped

Figure 26: Stopping my instance for the day.

When I started my instance up again the following day is when I was able to finish the remaining experimenting with the security group settings demonstrated throughout step 4. I had begun with widening permissions and slowly removing them to see what worked and what did not. It was very difficult trying to identify if I was completing the steps correctly because I ran into an issue where I was changing too many settings at once, then having to restart a lot of times because I was unsure of myself.

The following day when I went back to review my logs to check which target ports were attempted to be accessed, I could see a couple from the initial time I successfully ran a **status.sh**. I could also notice that my public IP address was showing in a lot of the source IP addresses displayed in the screenshot below because of the inbound permissions I had given it, but also noticed other IP addresses some with different risk levels and different ports they wanted to access, most of them low risk level.

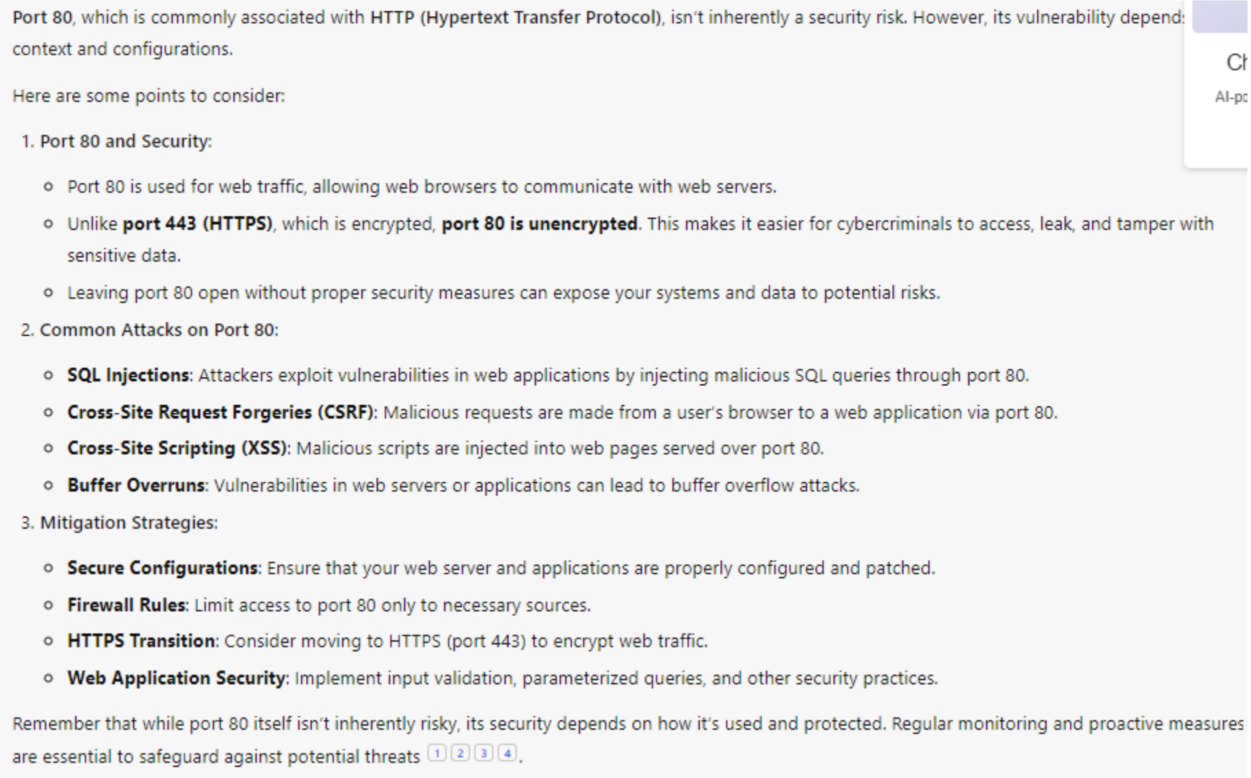


Date	Time	Source	Source Port	Target	Target Port	Proto.
2024-04-28	22:54:14	218.92.0.99	9090	172.31.37.71	22	6
2024-04-28	23:01:10	163.123.72.218	62989	172.31.37.71	22	6
2024-04-28	23:10:35	163.123.72.218	63060	172.31.37.71	22	6
2024-04-28	23:36:54	80.75.212.75	34778	172.31.37.71	13785	6
2024-04-28	23:36:54	162.216.150.177	55182	172.31.37.71	62548	6
2024-04-28	23:36:55	52.80.72.65	8	172.31.37.71	0	1
2024-04-28	23:37:11	45.56.84.110	58720	172.31.37.71	30287	6
2024-04-28	23:37:14	96.44.110.1	30369	172.31.37.71	23	6
2024-04-28	23:37:14	96.44.110.1	40546	172.31.37.71	23	6
2024-04-28	23:38:05	162.216.150.152	51048	172.31.37.71	45992	6
2024-04-28	23:38:06	163.123.72.218	63478	172.31.37.71	12222	6
2024-04-28	23:38:07	163.123.72.218	63478	172.31.37.71	12222	6

Figure 27: Checking Logging from the first day installing DShield.

Some logs that I reviewed warned me that the IP being used was for Internet Wide research scans which I thought was interesting. I ran into another IP address **66.35.60.2** on my list of logs that had a risk of 5 and target port 80 so with this information I could easily click on the address, scroll down, and see the "Whois" information and there it included a NetName, OrgName, OrgTechName, and so much more information. With this, I decided to see if this is a common targeted port for attacks, so I asked Microsoft CoPilot.

My question was "So i am reviewing my DShield log and want to see if port 80 is a commonly targeted port used for attacks? I have seen it a few times from the same IP address." In the figure below you can see that this port is not inherently a security risk, but it can pose a vulnerability due to the fact that it is an unencrypted port and this can lead cybercriminals to access, leak and tamper with sensitive data. I was also able to learn some mitigation strategies from this by looking more into port 80, in that it is important to setup secure configurations, limiting access on port 80, and allowing HTTPS for web traffic at all times. The practice of allowing HTTP traffic was done earlier in this lab [3].



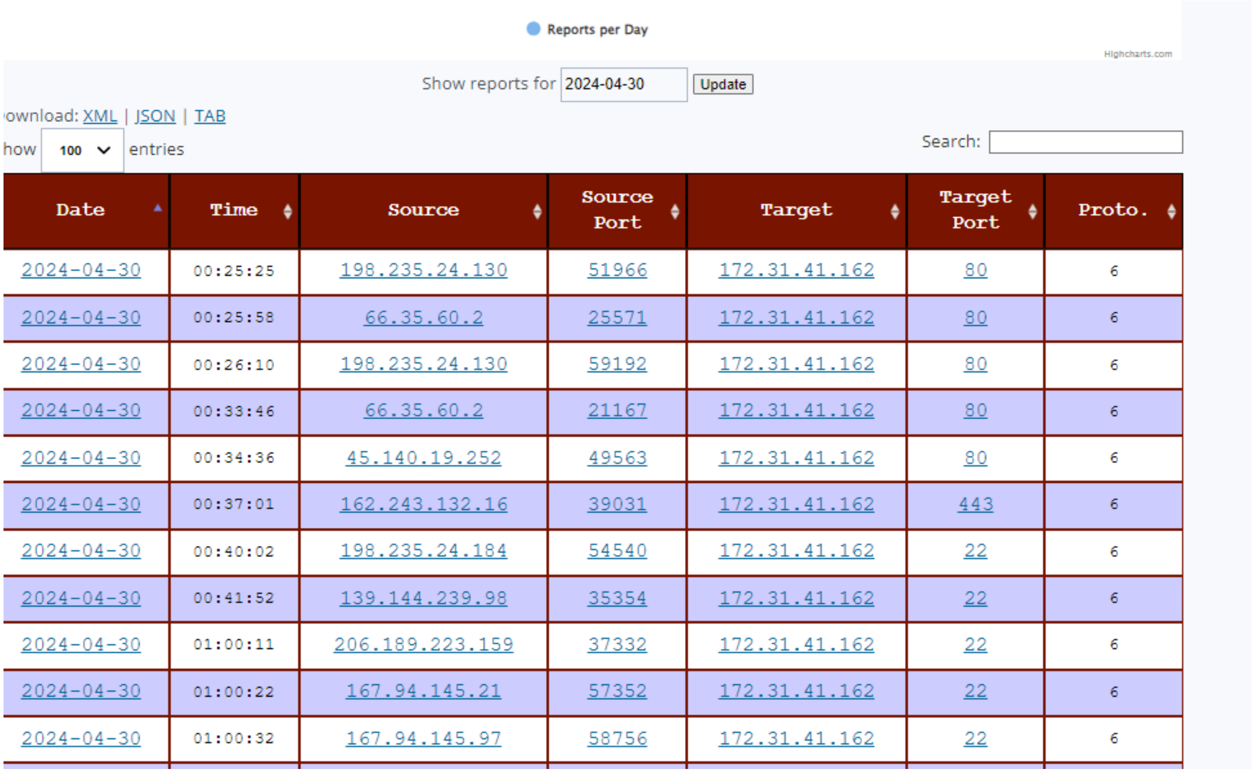
Port 80, which is commonly associated with **HTTP (Hypertext Transfer Protocol)**, isn't inherently a security risk. However, its vulnerability depend context and configurations.

Here are some points to consider:

1. **Port 80 and Security**:
    - Port 80 is used for web traffic, allowing web browsers to communicate with web servers.
    - Unlike **port 443 (HTTPS)**, which is encrypted, **port 80 is unencrypted**. This makes it easier for cybercriminals to access, leak, and tamper with sensitive data.
    - Leaving port 80 open without proper security measures can expose your systems and data to potential risks.
2. **Common Attacks on Port 80**:
    - **SQL Injections**: Attackers exploit vulnerabilities in web applications by injecting malicious SQL queries through port 80.
    - **Cross-Site Request Forgeries (CSRF)**: Malicious requests are made from a user's browser to a web application via port 80.
    - **Cross-Site Scripting (XSS)**: Malicious scripts are injected into web pages served over port 80.
    - **Buffer Overruns**: Vulnerabilities in web servers or applications can lead to buffer overflow attacks.
3. **Mitigation Strategies**:
    - **Secure Configurations**: Ensure that your web server and applications are properly configured and patched.
    - **Firewall Rules**: Limit access to port 80 only to necessary sources.
    - **HTTPS Transition**: Consider moving to HTTPS (port 443) to encrypt web traffic.
    - **Web Application Security**: Implement input validation, parameterized queries, and other security practices.

Remember that while port 80 itself isn't inherently risky, its security depends on how it's used and protected. Regular monitoring and proactive measures are essential to safeguard against potential threats [1] [2] [3] [4].

Figure 28: Learning about port 80 and the vulnerabilities it poses.



Show reports for 2024-04-30

ownload: XML | JSON | TAB

how 100 entries

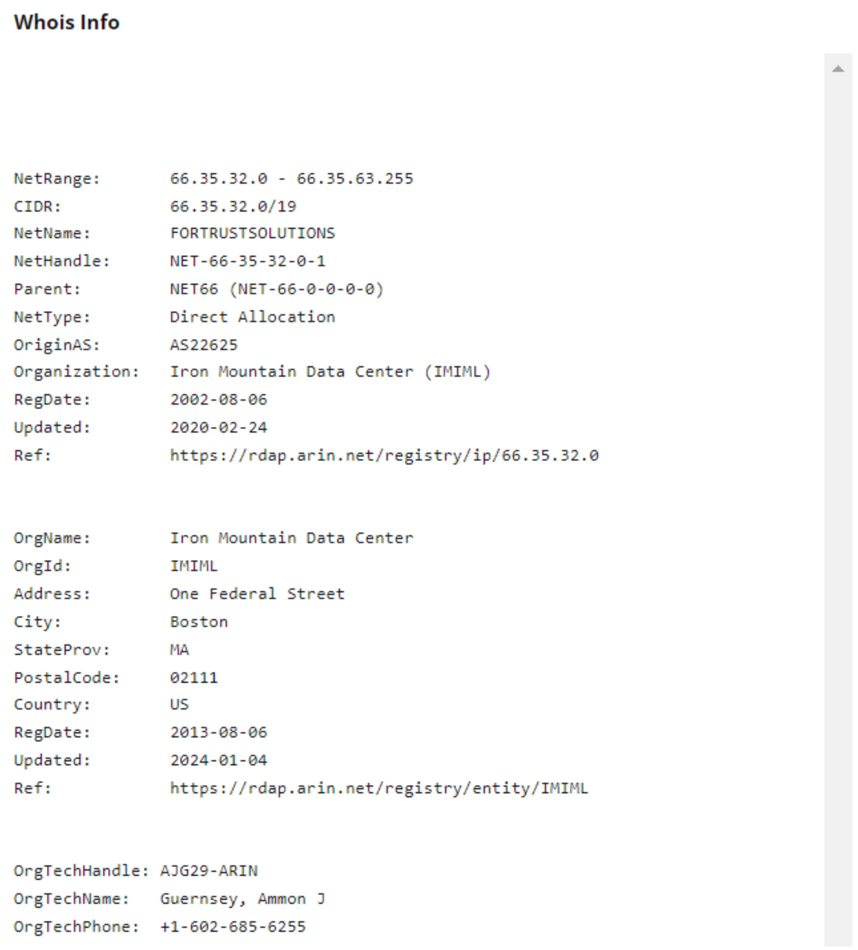
Search:

Date	Time	Source	Source Port	Target	Target Port	Proto.
2024-04-30	00:25:25	198.235.24.130	51966	172.31.41.162	80	6
2024-04-30	00:25:58	66.35.60.2	25571	172.31.41.162	80	6
2024-04-30	00:26:10	198.235.24.130	59192	172.31.41.162	80	6
2024-04-30	00:33:46	66.35.60.2	21167	172.31.41.162	80	6
2024-04-30	00:34:36	45.140.19.252	49563	172.31.41.162	80	6
2024-04-30	00:37:01	162.243.132.16	39031	172.31.41.162	443	6
2024-04-30	00:40:02	198.235.24.184	54540	172.31.41.162	22	6
2024-04-30	00:41:52	139.144.239.98	35354	172.31.41.162	22	6
2024-04-30	01:00:11	206.189.223.159	37332	172.31.41.162	22	6
2024-04-30	01:00:22	167.94.145.21	57352	172.31.41.162	22	6
2024-04-30	01:00:32	167.94.145.97	58756	172.31.41.162	22	6

Figure 29: Checking more logs on DShield.

Down below I have also included a screenshot of partial “WhoIs” information given by details on IP **66.35.60.2**.

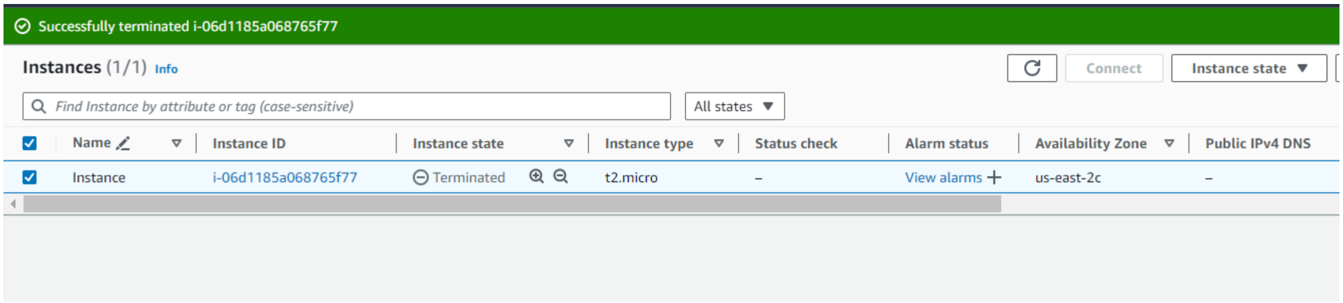
After I was done with my instance, I also made sure to terminate it as shown in figure 31 below.



Whois Info

```
NetRange:       66.35.32.0 - 66.35.63.255
CIDR:           66.35.32.0/19
NetName:        FORTRUSTSOLUTIONS
NetHandle:      NET-66-35-32-0-1
Parent:         NET66 (NET-66-0-0-0-0)
NetType:        Direct Allocation
OriginAS:       AS22625
Organization:   Iron Mountain Data Center (IMIML)
RegDate:        2002-08-06
Updated:        2020-02-24
Ref:            https://rdap.arin.net/registry/ip/66.35.32.0


OrgName:        Iron Mountain Data Center
OrgId:          IMIML
Address:        One Federal Street
City:           Boston
StateProv:      MA
PostalCode:     02111
Country:        US
RegDate:        2013-08-06
Updated:        2024-01-04
Ref:            https://rdap.arin.net/registry/entity/IMIML


OrgTechHandle: AJG29-ARIN
OrgTechName:   Guernsey, Ammon J
OrgTechPhone:  +1-602-685-6255
```

Figure 30: WhoIs information given by one of logs scanned on port 80.



Successfully terminated i-06d1185a068765f77

Instances (1/1) Info

☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☑	Instance	i-06d1185a068765f77	Terminated	t2.micro	–	View alarms +	us-east-2c	–

Figure 31: Successfully terminating my instance.

CONCLUSION

To conclude this lab, I was able to take lots of notes on how to deploy a honeypot for the next time around. As I continued to reread instructions and find more resources to compare my experience, I was able to take feedback and use it to guide me through the lab, mostly when attempting to run the status.sh and access logs. Through this research project we participated in I was able to take away the significance in the role honeypots play in cybersecurity. They allow us to study tactics that attackers may use, techniques and procedures to prevent attacks from happening further down the road. This was a great way of warming up to honeypots and having the desire to explore more about how they can be used as an asset in the cybersecurity industry.

REFERENCES

[1] M. OBrien, “Setting Up A DShield Honeypot In AWS,” Medium. Accessed: Apr. 25, 2024. [Online]. Available: https://matthewob5.medium.com/setting-up-a-dshield-honeypot-in-aws-2ca5f8a29d9

[2] “what if i am trying to ssh through port 12222 into a cloud server from my ip address, but i am not able to because the ip address is incorrect?” prompt. *ChatGPT*, Version 3.5, 29 Apr. 2024, chat.openai.com/chat.

[3] Microsoft Copilot [GPT-4 language model], response to “So i am reviewing my DShield log and want to see if port 80 is a commonly targeted port used for attacks? I have seen it a few times from the same IP address.”. Microsoft & OpenAI *CoPilot*, *OpenAI Chatbot, 30 Apr. 2024*, https://copilot.microsoft.com/.

[4] “Dshield/Docs/Install-Instructions/AWS.Md at Main · DShield-ISC/Dshield.” *GitHub*, https://github.com/DShield-ISC/dshield/blob/main/docs/install-instructions/AWS.md. Accessed 28 Apr. 2024.

[5] Mitra, Department of Information Systems and Cyber Security. [Online] “Lab-08 Deployiing a Honeypot” by Rita Mitra (2024). UTSA, 2024 [Accessed: April 23, 2024.]

COLLABORATION

In this lab, I cross referenced between different instructions provided for deploying honeypots with DShield shown above, including the professor’s lab instructions, as well as the Matthew O’Brien blog post. I was able to use our class Slack module 08 channel to see if my peers were encountering the same problems as I was, and this was very helpful in my learning. I also used generative AI tools to help answer questions I had.