Modern Honeypot Network

Final Project Report

CMPE 132- Information Security

Professor Chao-Li Tarng

Project Group 5

Ryan Biesty, Danielle Shen, and Ryota Suzuki

December 7th, 2020

Project Overview

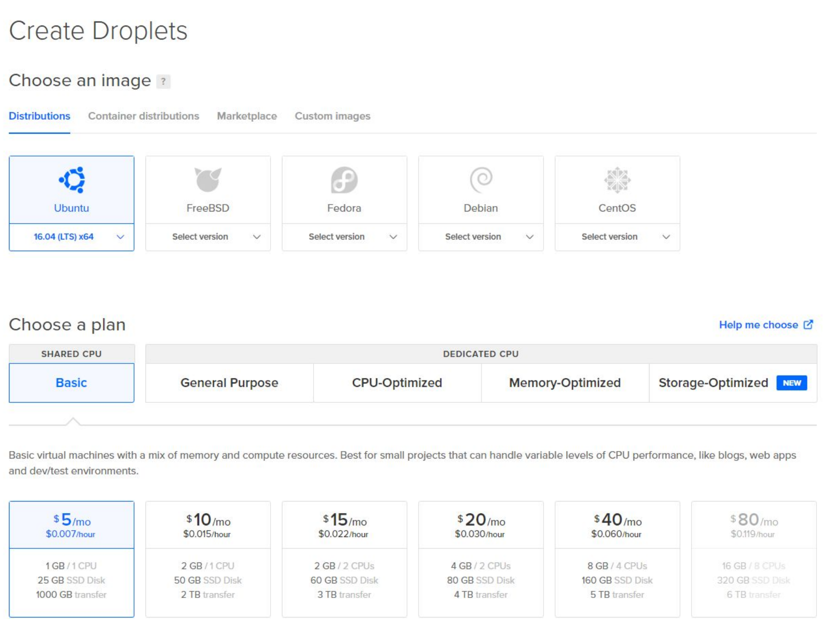
This project is the Modern Honeypot Network (MHN) to determine the most frequent attacks that occur on a network. This is to better understand how and how frequently attacks are carried out in our day-to-day lives. It is difficult to protect against attacks if one does not know the attacker. Although these MHN are not a network security solution, it allows the defenders to better understand where weak points may be in the network.

Objective and Scope

Honeypots are sometimes used as decoys for true servers. In this project, our goal is to demonstrate and analyze the Modern Honeypot Network by setting up a fake server and observing the attacks that occur. To do this, the honeypot must look somewhat legitimate to receive attacks not only from ourselves. By observing the attacks on the MHN server will help us draw a clearer picture of our attackers and the methods they use to attempt to breach.

Set up

In order to set up a MHN with honeypots, first we need to create a MHN server. To do this, we borrowed a digital server from a digitalocean. We decided to borrow a server instead of making a VM for it since we would need to keep a device on for 24/7, and also for security purposes since this is going to track the attention of hackers. The first step to creating this server is to set up what they call a "droplet". We recommend using Ubuntu 16.04 since the setup process requires python2, and the newest Ubuntu version currently available (20.04) supports python3. We also chose the basic plan since we do not need much power nor space. Then, choose a datacenter location that is relatively close to you so it does not take too long to connect to it. Finally, create a password to login to the MHN server and name it something like 'MHNServer' to easily distinguish between other droplets. The set up for the droplet is shown below.



Create Droplets

Choose an image ?

Distributions | Container distributions | Marketplace | Custom images

Ubuntu	FreeBSD	Fedora	Debian	CentOS
16.04 (LTS) x64	Select version	Select version	Select version	Select version

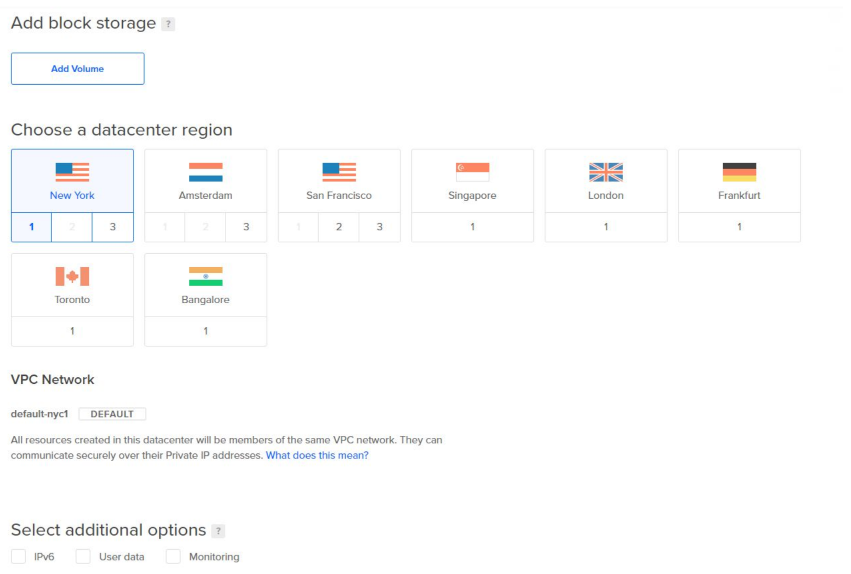
Choose a plan — Help me choose

SHARED CPU	DEDICATED CPU			
Basic	General Purpose	CPU-Optimized	Memory-Optimized	Storage-Optimized NEW

Basic virtual machines with a mix of memory and compute resources. Best for small projects that can handle variable levels of CPU performance, like blogs, web apps and dev/test environments.

$5/mo	$10/mo	$15/mo	$20/mo	$40/mo	$80/mo
$0.007/hour	$0.015/hour	$0.022/hour	$0.030/hour	$0.060/hour	$0.119/hour
1 GB / 1 CPU	2 GB / 1 CPU	2 GB / 2 CPUs	4 GB / 2 CPUs	8 GB / 4 CPUs	16 GB / 8 CPUs
25 GB SSD Disk	50 GB SSD Disk	60 GB SSD Disk	80 GB SSD Disk	160 GB SSD Disk	320 GB SSD Disk
1000 GB transfer	2 TB transfer	3 TB transfer	4 TB transfer	5 TB transfer	6 TB transfer

Figure 1: Digitalocean Droplet Setup



Add block storage ?

Add Volume

Choose a datacenter region

New York	Amsterdam	San Francisco	Singapore	London	Frankfurt	Toronto	Bangalore
1 2 3	1 2 3	1 2 3	1	1	1	1	1

VPC Network

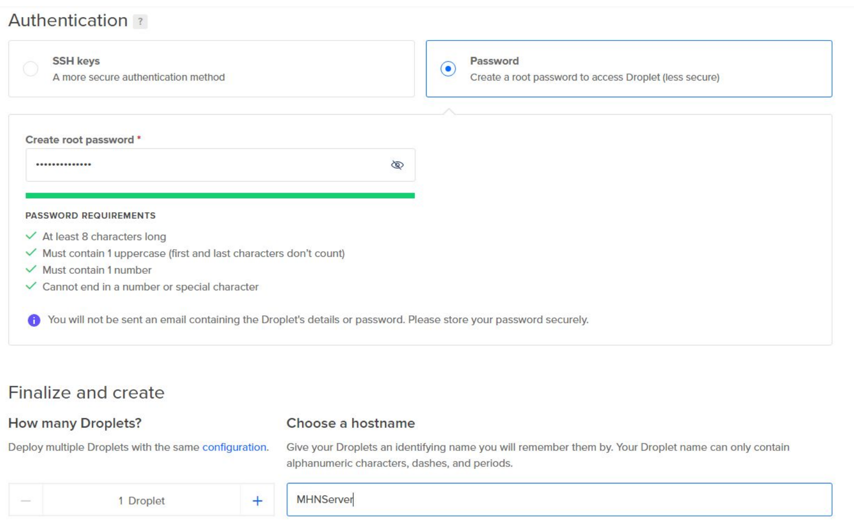
default-nyc1 DEFAULT

All resources created in this datacenter will be members of the same VPC network. They can communicate securely over their Private IP addresses. What does this mean?

Select additional options ?

☐ IPv6 ☐ User data ☐ Monitoring

Figure 2: Digitalocean Droplet Setup (cont.)


Authentication

SSH keys
A more secure authentication method

Password
Create a root password to access Droplet (less secure)

Create root password *
••••••••••••••

PASSWORD REQUIREMENTS
- ✓ At least 8 characters long
- ✓ Must contain 1 uppercase (first and last characters don't count)
- ✓ Must contain 1 number
- ✓ Cannot end in a number or special character

You will not be sent an email containing the Droplet's details or password. Please store your password securely.

Finalize and create

How many Droplets?
Deploy multiple Droplets with the same configuration.

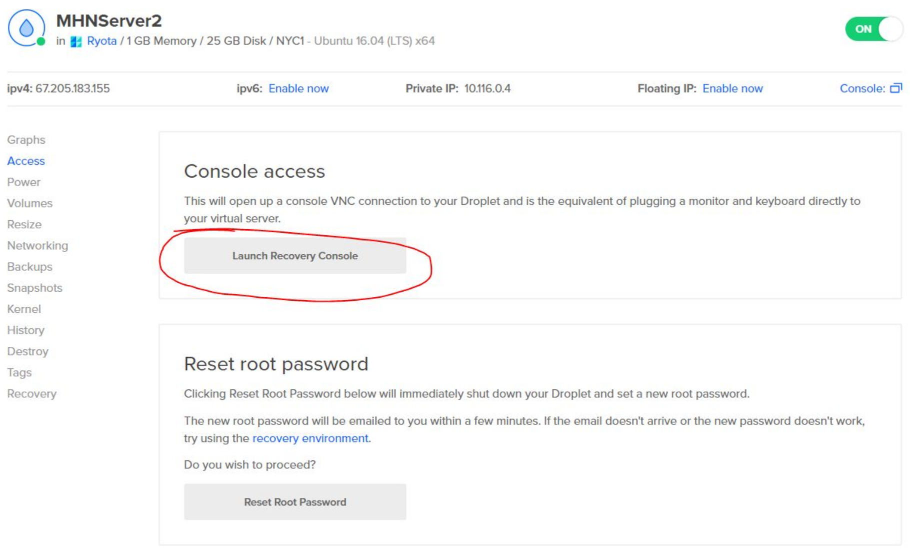
1 Droplet

Choose a hostname
Give your Droplets an identifying name you will remember them by. Your Droplet name can only contain alphanumeric characters, dashes, and periods.

MHNServer

Figure 3: Digitalocean Droplet Setup (cont.)

After creating a droplet, open the MHN server and click the "Launch Recovery Console" under MHNServer -> Access.


MHNServer2
in Ryota / 1 GB Memory / 25 GB Disk / NYC1 - Ubuntu 16.04 (LTS) x64

ON

ipv4: 67.205.183.155 ipv6: Enable now Private IP: 10.116.0.4 Floating IP: Enable now Console:

Console access
This will open up a console VNC connection to your Droplet and is the equivalent of plugging a monitor and keyboard directly to your virtual server.

Launch Recovery Console

Reset root password
Clicking Reset Root Password below will immediately shut down your Droplet and set a new root password.

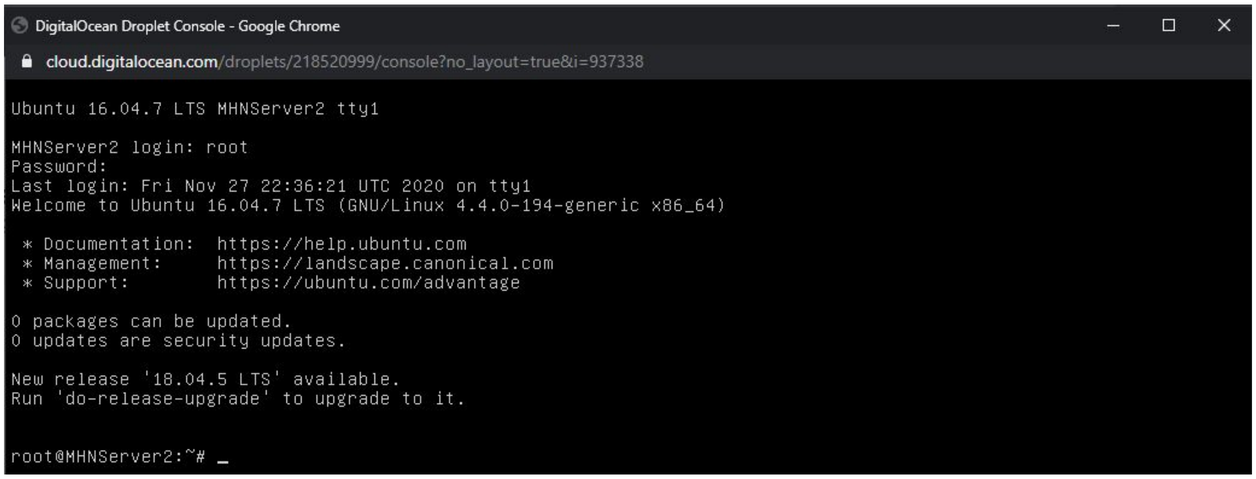
The new root password will be emailed to you within a few minutes. If the email doesn't arrive or the new password doesn't work, try using the recovery environment.

Do you wish to proceed?

Reset Root Password

Figure 4: Digitalocean MHN Setup

Then, a console should appear in your default browser asking for login. Username of the server is 'root' at default, and password is the one you set earlier in the droplet set up. After a successful login, the console should look something like this.


DigitalOcean Droplet Console - Google Chrome
cloud.digitalocean.com/droplets/218520999/console?no_layout=true&i=937338

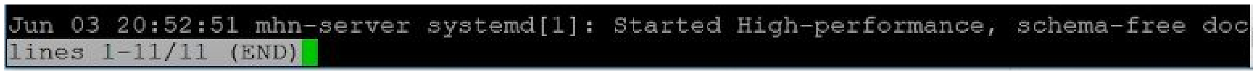
```
Ubuntu 16.04.7 LTS MHNServer2 tty1

MHNServer2 login: root
Password:
Last login: Fri Nov 27 22:36:21 UTC 2020 on tty1
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-194-generic x86_64)

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

0 packages can be updated.
0 updates are security updates.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


root@MHNServer2:~# _
```

Figure 5: Digitalocean MHN Setup Terminal

Next, we set up the MHN server with the following command.

```
$ cd /opt/
$ sudo git clone https://github.com/pwnlandia/mhn.git
$ cd mhn/
$ sudo ./install.sh
```

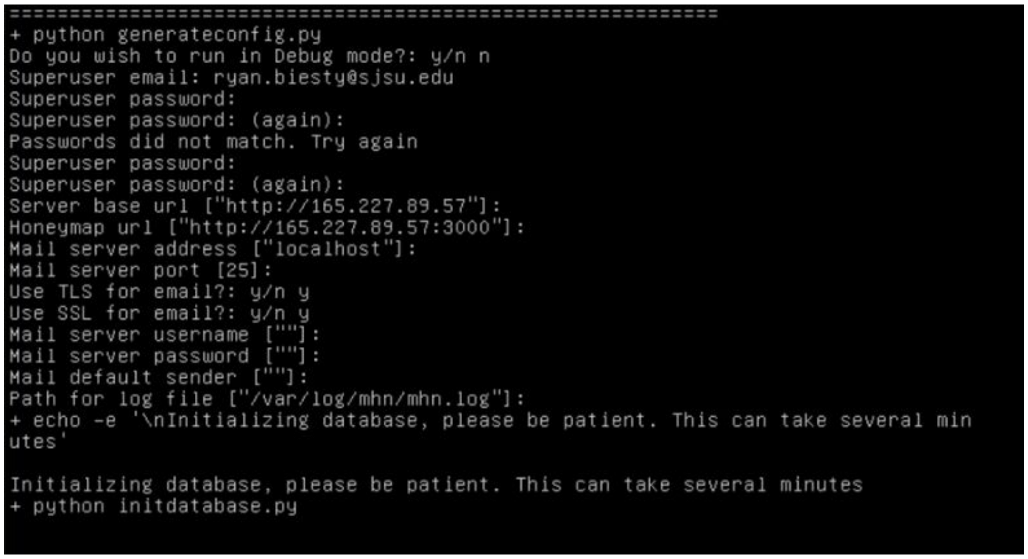
After sending the last command, it will take a couple minutes to finish installing. If a screen like this comes in, press 'q' and the installation will continue.



```
Jun 03 20:52:51 mhn-server systemd[1]: Started High-performance, schema-free doc
lines 1-11/11 (END)
```

Figure 6: Digitalocean MHN Setup Terminal (cont.)

After a successful installation, the console will run you through the configuration settings. Follow the configuration setting as the screenshot below, with your email and password. The superuser email and password are going to be used to later login to the MHN server web application.



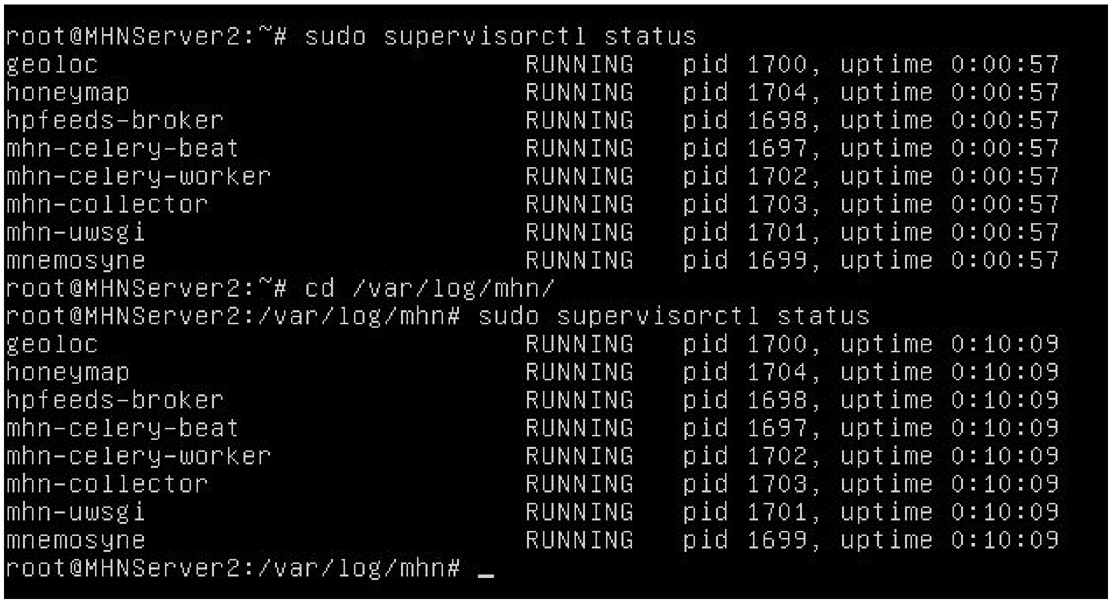
```
==========================================================
+ python generateconfig.py
Do you wish to run in Debug mode?: y/n n
Superuser email: ryan.biesty@sjsu.edu
Superuser password:
Superuser password: (again):
Passwords did not match. Try again
Superuser password:
Superuser password: (again):
Server base url ["http://165.227.89.57"]:
Honeymap url ["http://165.227.89.57:3000"]:
Mail server address ["localhost"]:
Mail server port [25]:
Use TLS for email?: y/n y
Use SSL for email?: y/n y
Mail server username [""]:
Mail server password [""]:
Mail default sender [""]:
Path for log file ["/var/log/mhn/mhn.log"]:
+ echo -e '\nInitializing database, please be patient. This can take several min
utes'

Initializing database, please be patient. This can take several minutes
+ python initdatabase.py
```

Figure 7: Digitalocean MHN Setup Terminal (cont.)

Wait a couple minutes for this to finish installing. To see if everything is running, type the following command. This is what you should see if everything is running correctly.

$ cd /var/log/mhn/

$ sudo supervisorctl status



```
root@MHNServer2:~# sudo supervisorctl status
geoloc                           RUNNING   pid 1700, uptime 0:00:57
honeymap                         RUNNING   pid 1704, uptime 0:00:57
hpfeeds-broker                   RUNNING   pid 1698, uptime 0:00:57
mhn-celery-beat                  RUNNING   pid 1697, uptime 0:00:57
mhn-celery-worker                RUNNING   pid 1702, uptime 0:00:57
mhn-collector                    RUNNING   pid 1703, uptime 0:00:57
mhn-uwsgi                        RUNNING   pid 1701, uptime 0:00:57
mnemosyne                        RUNNING   pid 1699, uptime 0:00:57
root@MHNServer2:~# cd /var/log/mhn/
root@MHNServer2:/var/log/mhn# sudo supervisorctl status
geoloc                           RUNNING   pid 1700, uptime 0:10:09
honeymap                         RUNNING   pid 1704, uptime 0:10:09
hpfeeds-broker                   RUNNING   pid 1698, uptime 0:10:09
mhn-celery-beat                  RUNNING   pid 1697, uptime 0:10:09
mhn-celery-worker                RUNNING   pid 1702, uptime 0:10:09
mhn-collector                    RUNNING   pid 1703, uptime 0:10:09
mhn-uwsgi                        RUNNING   pid 1701, uptime 0:10:09
mnemosyne                        RUNNING   pid 1699, uptime 0:10:09
root@MHNServer2:/var/log/mhn# _
```

Figure 8: Digitalocean MHN Setup Terminal (cont.)

Now that the MHN server is up and running, open the MHN server with the IP address of it. To find the IP address of the MHN server, go to digital ocean's server page. If you type 'ifconfig' to the console to get the IP address, you get the private IP address of the server. This will not work, so do not use that.

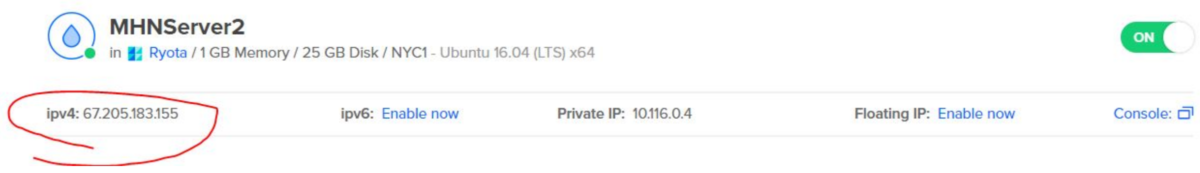


Figure 9: Digitalocean MHN Setup Terminal (cont.)

Then, type the IP address in your web browser and you will see the following page. Login with your superuser email and password you set up in the MHN configuration.

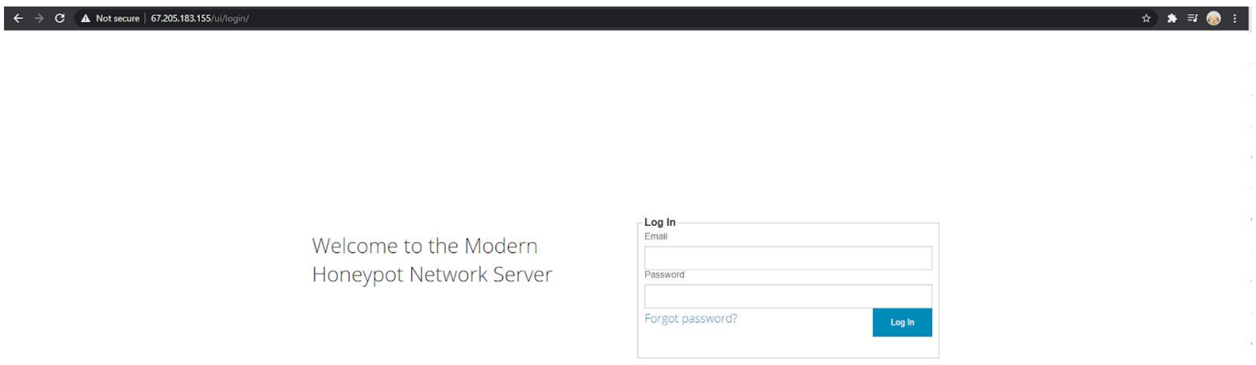
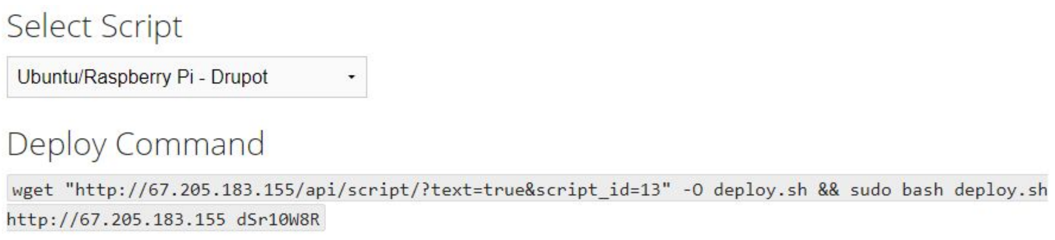

Not secure | 67.205.183.155/ui/login/
Welcome to the Modern Honeypot Network Server
Log In
Email
Password
Forgot password?
Log In

Figure 10: Digitalocean MHN Setup Terminal (cont.)

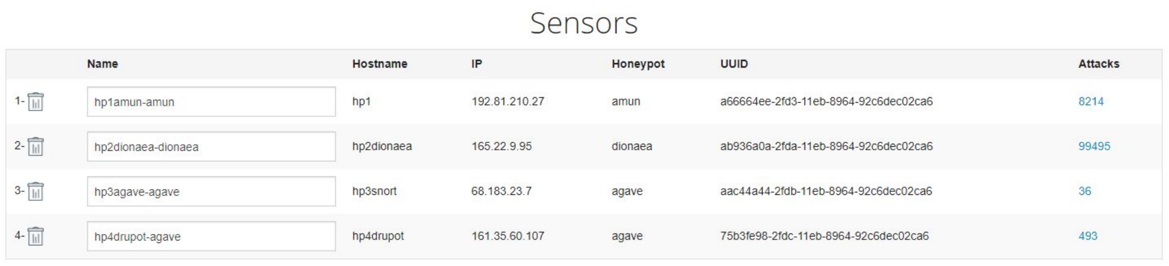
Here you will be able to see all attacks on your honeypots with the attacker's IP, ports attacked, time, etc. After you log in, hit deploy and select a script. Not all of them are supported/up to date so choose a one that works. From our testing, drupot, agave, dionaea, and amun were successful and easy to set up. After selecting a script, a deploy command should show as below. Copy it.



```
wget "http://67.205.183.155/api/script/?text=true&script_id=13" -O deploy.sh && sudo bash deploy.sh
http://67.205.183.155 dSr10W8R
```

Figure 11: Digitalocean MHN Setup Terminal (cont.)

Now we will now set up our honeypots. Create another droplet as previously done with the MHN server, but with a different name like ‘Honeypot1’. Then open it with the launch recovery console and log in. Now, all we have to do is paste the deploy code and the honeypot is up. If successful, you go back to our MHN server and hit sensors, you will see your honeypot show up.



Sensors

	Name	Hostname	IP	Honeypot	UUID	Attacks
1-	hp1amun-amun	hp1	192.81.210.27	amun	a66664ee-2fd3-11eb-8964-92c6dec02ca6	8214
2-	hp2dionaea-dionaea	hp2dionaea	165.22.9.95	dionaea	ab936a0a-2fda-11eb-8964-92c6dec02ca6	99495
3-	hp3agave-agave	hp3snort	68.183.23.7	agave	aac44a44-2fdb-11eb-8964-92c6dec02ca6	36
4-	hp4drupot-agave	hp4drupot	161.35.60.107	agave	75b3fe98-2fdc-11eb-8964-92c6dec02ca6	493

Figure 12: Digitalocean MHN Setup Terminal (cont.)

You can set up multiple honeypots with the MHN server. To do this, make a new droplet with another name, and run the deploy command. They should all appear on the sensor page when successful. Now wait and observe the attacks that come in.

Architecture Design

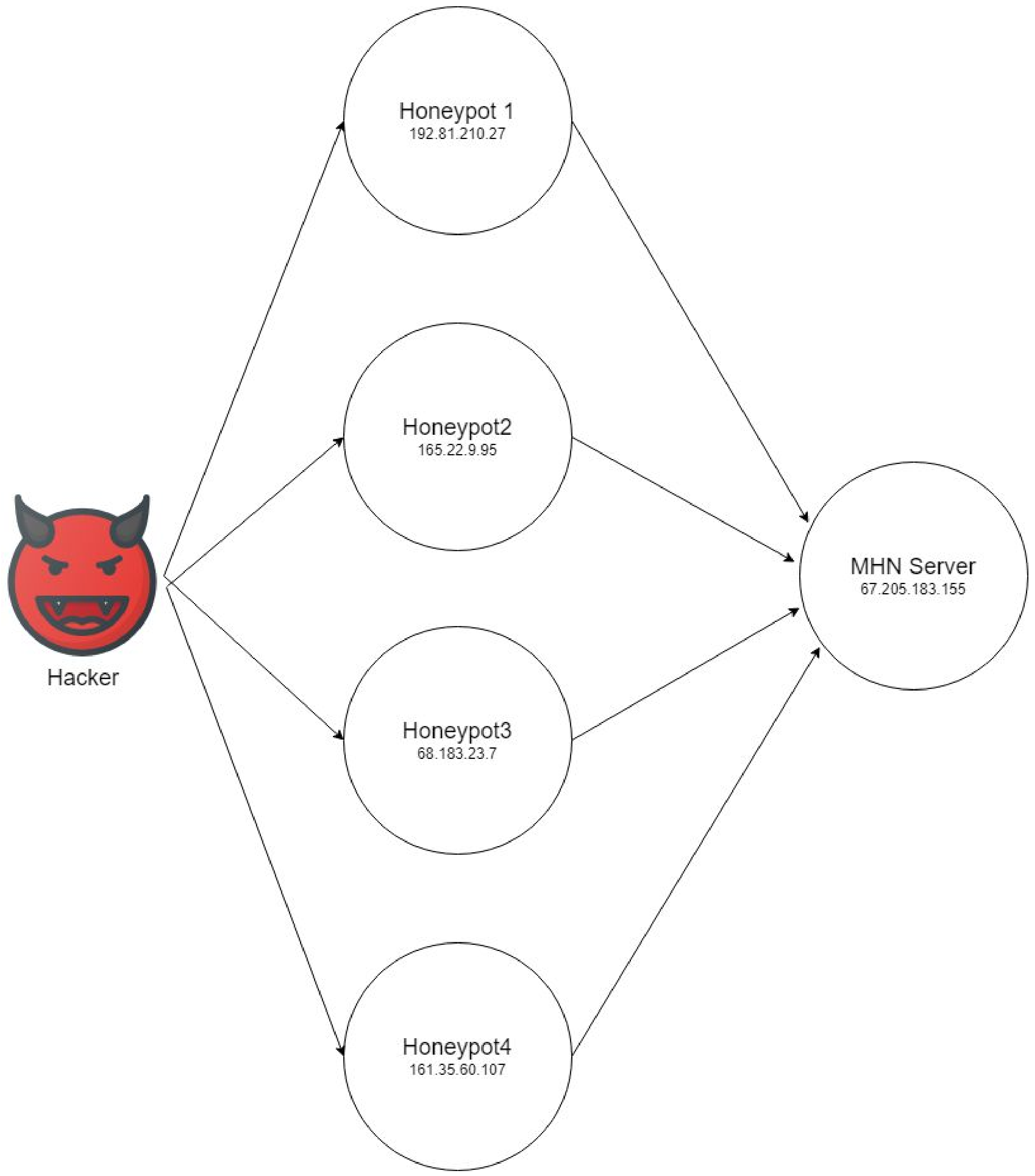
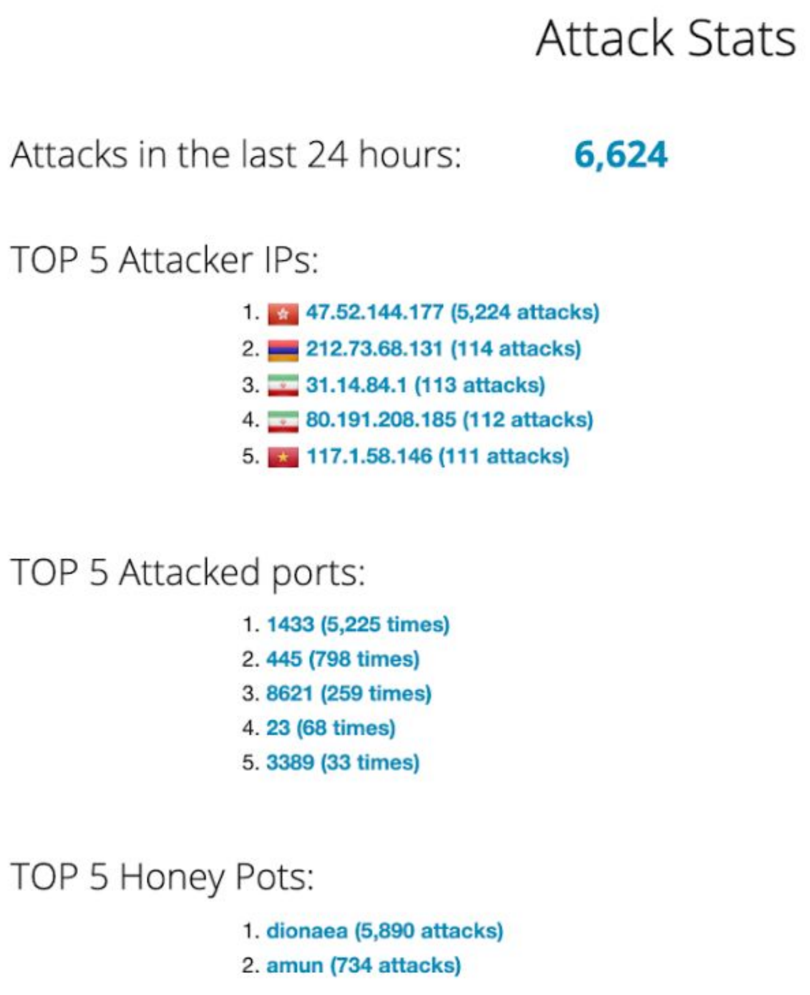


Figure 13: Diagram for MHN Architecture

Results and Analysis

After setting up both the MHN and honeypots, we waited 24 hours to collect our data. To have a bigger sample size for better data, we ran this test multiple times with different MHN and honeypots. Our first MHN crashed as the log memory file got too big, causing us to restart our data collection. For our data, we were able to collect the ID address of the attacker, the port they attacked, as well as which honeypot was attacked. The MHN server was able to collect these data and create a threat map to better visualize where the attacks came from. The collected data are shown below.



Attack Stats

Attacks in the last 24 hours: 6,624

TOP 5 Attacker IPs:

1. 47.52.144.177 (5,224 attacks)
2. 212.73.68.131 (114 attacks)
3. 31.14.84.1 (113 attacks)
4. 80.191.208.185 (112 attacks)
5. 117.1.58.146 (111 attacks)

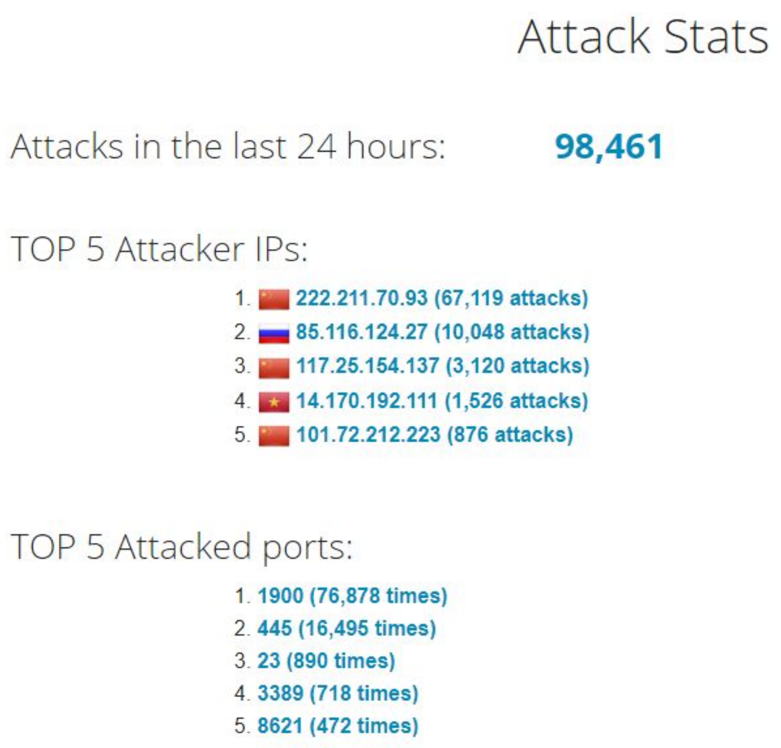
TOP 5 Attacked ports:

1. 1433 (5,225 times)
2. 445 (798 times)
3. 8621 (259 times)
4. 23 (68 times)
5. 3389 (33 times)

TOP 5 Honey Pots:

1. dionaea (5,890 attacks)
2. amun (734 attacks)

Figure 14: Attacks logs on MHN from Nov25-26th



Attack Stats

Attacks in the last 24 hours: **98,461**

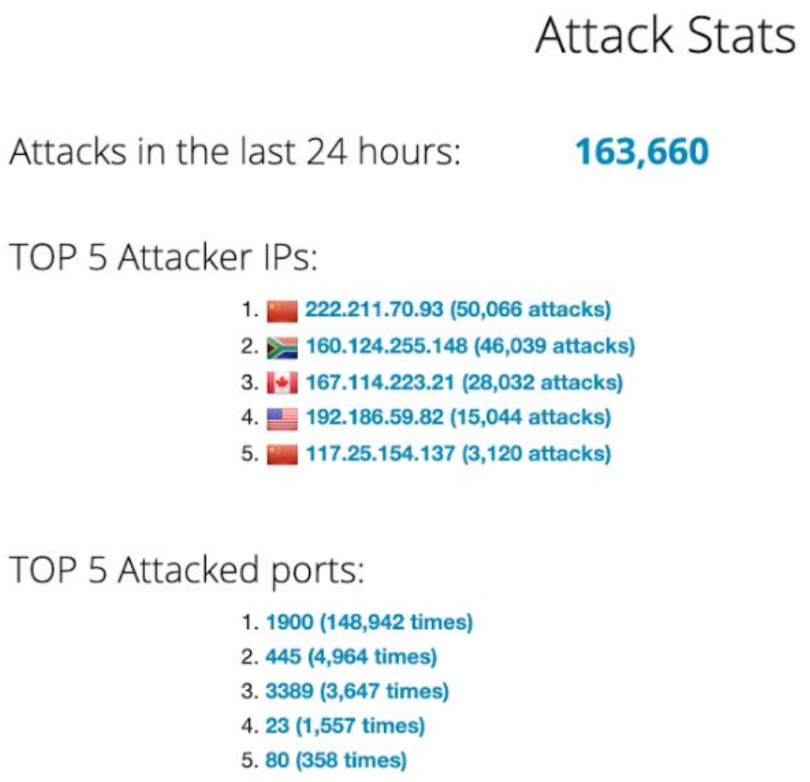
TOP 5 Attacker IPs:

1. 222.211.70.93 (67,119 attacks)
2. 85.116.124.27 (10,048 attacks)
3. 117.25.154.137 (3,120 attacks)
4. 14.170.192.111 (1,526 attacks)
5. 101.72.212.223 (876 attacks)

TOP 5 Attacked ports:

1. 1900 (76,878 times)
2. 445 (16,495 times)
3. 23 (890 times)
4. 3389 (718 times)
5. 8621 (472 times)

Figure 15: Attacks logs on MHN from Nov26-27th



Attack Stats

Attacks in the last 24 hours: **163,660**

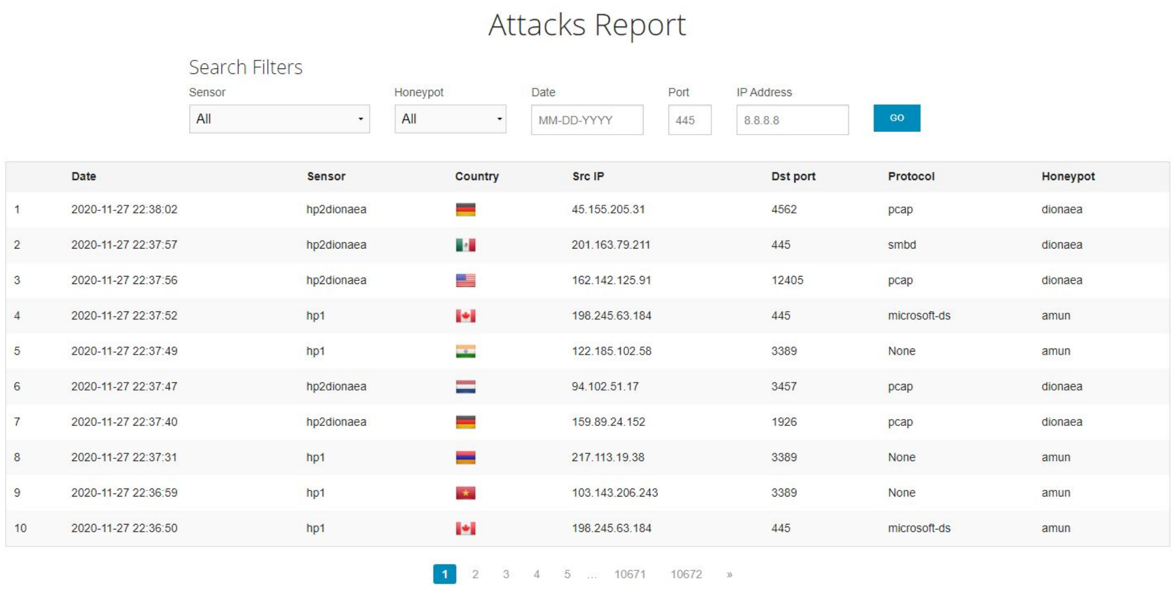
TOP 5 Attacker IPs:

1. 222.211.70.93 (50,066 attacks)
2. 160.124.255.148 (46,039 attacks)
3. 167.114.223.21 (28,032 attacks)
4. 192.186.59.82 (15,044 attacks)
5. 117.25.154.137 (3,120 attacks)

TOP 5 Attacked ports:

1. 1900 (148,942 times)
2. 445 (4,964 times)
3. 3389 (3,647 times)
4. 23 (1,557 times)
5. 80 (358 times)

Figure 16: Attacks logs on MHN from Nov27-28th



Attacks Report

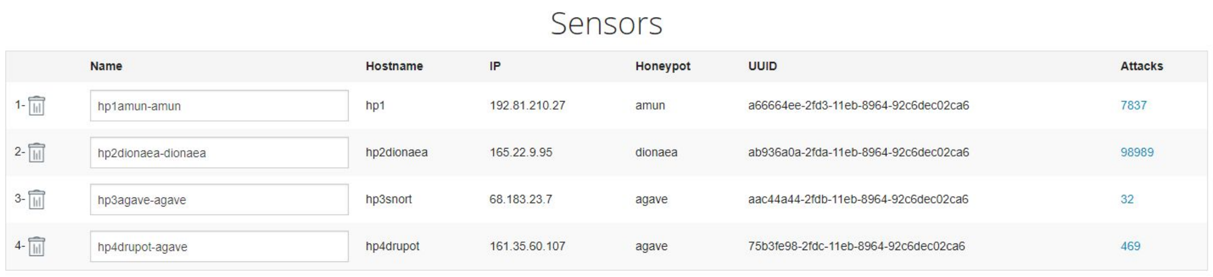
Search Filters

Sensor	Honeypot	Date	Port	IP Address	
All	All	MM-DD-YYYY	445	8.8.8.8	GO

	Date	Sensor	Country	Src IP	Dst port	Protocol	Honeypot
1	2020-11-27 22:38:02	hp2dionaea	🇩🇪	45.155.205.31	4562	pcap	dionaea
2	2020-11-27 22:37:57	hp2dionaea	🇲🇽	201.163.79.211	445	smbd	dionaea
3	2020-11-27 22:37:56	hp2dionaea	🇺🇸	162.142.125.91	12405	pcap	dionaea
4	2020-11-27 22:37:52	hp1	🇨🇦	198.245.63.184	445	microsoft-ds	amun
5	2020-11-27 22:37:49	hp1	🇮🇳	122.185.102.58	3389	None	amun
6	2020-11-27 22:37:47	hp2dionaea	🇳🇱	94.102.51.17	3457	pcap	dionaea
7	2020-11-27 22:37:40	hp2dionaea	🇩🇪	159.89.24.152	1926	pcap	dionaea
8	2020-11-27 22:37:31	hp1	🇦🇲	217.113.19.38	3389	None	amun
9	2020-11-27 22:36:59	hp1	🇻🇳	103.143.206.243	3389	None	amun
10	2020-11-27 22:36:50	hp1	🇨🇦	198.245.63.184	445	microsoft-ds	amun

1 2 3 4 5 … 10671 10672 »

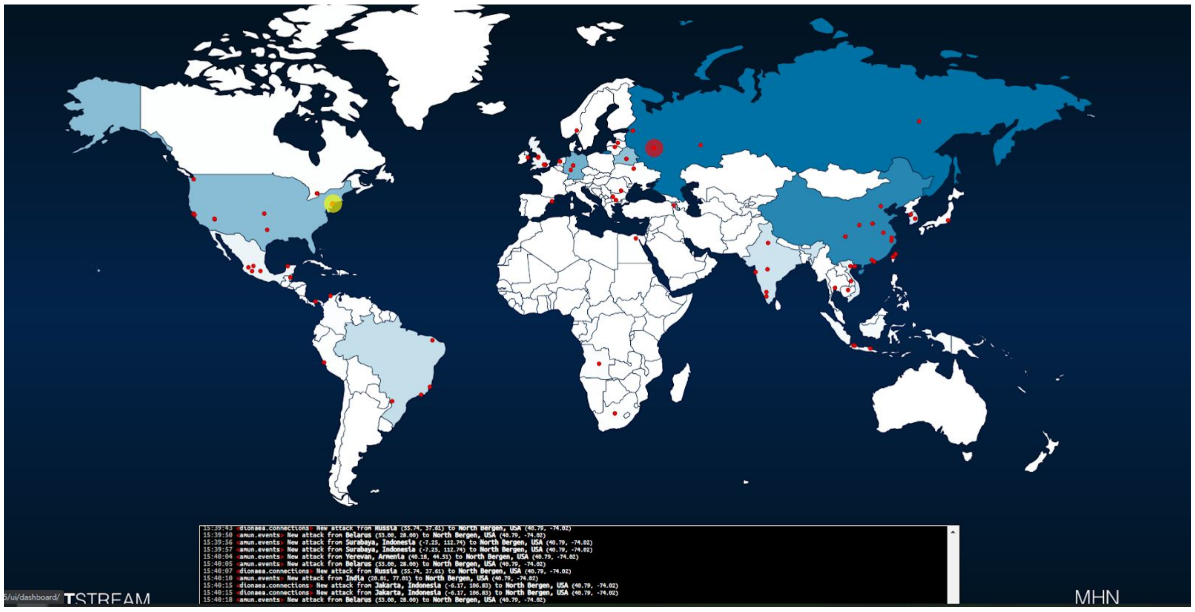
Figure 17: Sample attack reports from MHN



Sensors

	Name	Hostname	IP	Honeypot	UUID	Attacks
1-	hp1amun-amun	hp1	192.81.210.27	amun	a66664ee-2fd3-11eb-8964-92c6dec02ca6	7837
2-	hp2dionaea-dionaea	hp2dionaea	165.22.9.95	dionaea	ab936a0a-2fda-11eb-8964-92c6dec02ca6	98989
3-	hp3agave-agave	hp3snort	68.183.23.7	agave	aac44a44-2fdb-11eb-8964-92c6dec02ca6	32
4-	hp4drupot-agave	hp4drupot	161.35.60.107	agave	75b3fe98-2fdc-11eb-8964-92c6dec02ca6	469

Figure 18: Honeypots/sensors used



```
15:39:43 <dionaea.connections> New attack from Russia (55.74, 37.61) to North Bergen, USA (40.79, -74.02)
15:39:50 <amun.events> New attack from Belarus (53.00, 28.00) to North Bergen, USA (40.79, -74.02)
15:39:56 <amun.events> New attack from Surabaya, Indonesia (-7.25, 112.74) to North Bergen, USA (40.79, -74.02)
15:39:57 <amun.events> New attack from Surabaya, Indonesia (-7.25, 112.74) to North Bergen, USA (40.79, -74.02)
15:40:04 <amun.events> New attack from Yerevan, Armenia (40.18, 44.51) to North Bergen, USA (40.79, -74.02)
15:40:05 <amun.events> New attack from Belarus (53.00, 28.00) to North Bergen, USA (40.79, -74.02)
15:40:07 <dionaea.connections> New attack from Russia (55.74, 37.61) to North Bergen, USA (40.79, -74.02)
15:40:10 <amun.events> New attack from India (20.01, 77.01) to North Bergen, USA (40.79, -74.02)
15:40:15 <dionaea.connections> New attack from Jakarta, Indonesia (-6.17, 106.83) to North Bergen, USA (40.79, -74.02)
15:40:15 <dionaea.connections> New attack from Jakarta, Indonesia (-6.17, 106.83) to North Bergen, USA (40.79, -74.02)
15:40:18 <amun.events> New attack from Belarus (53.00, 28.00) to North Bergen, USA (40.79, -74.02)
```

Figure 19: Threat map

Taking these numbers into account, the information collected could be charted onto different graphics for a visual comparison. The following charts are from the first server created, about eight hours after the initial set up.

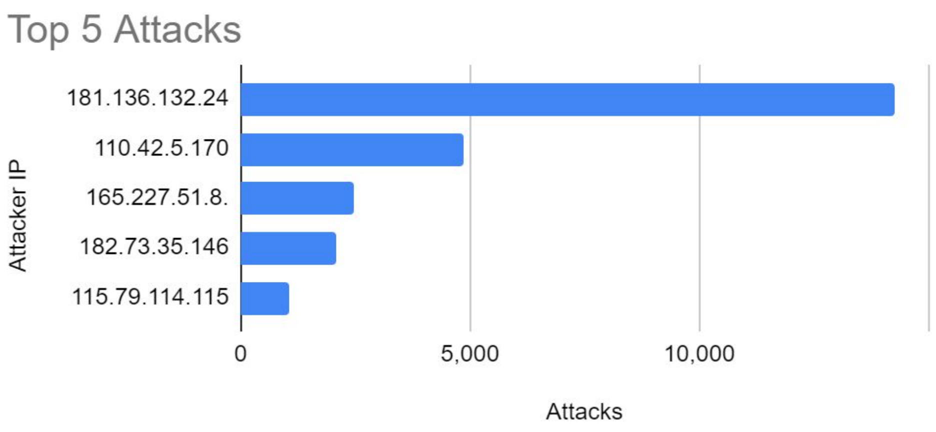


Figure 20: Top 5 attacker IP from Nov26th-28th

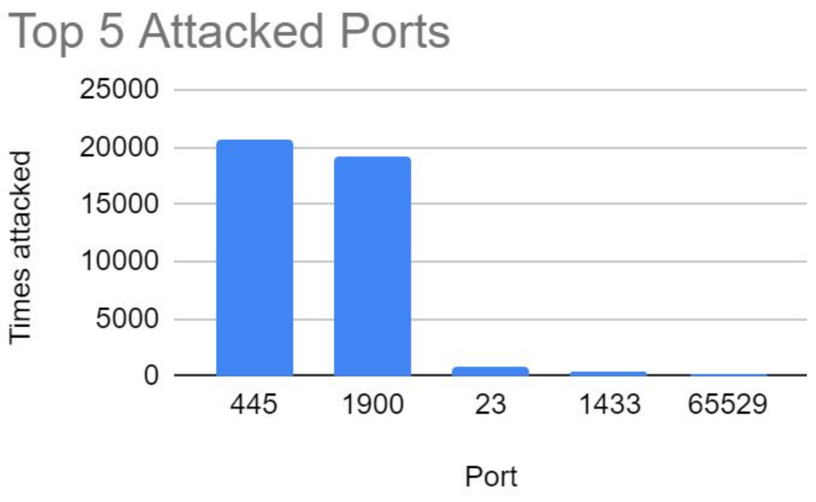


Figure 21: Top 5 ports attacked from Nov26-28th

Taking into consideration the differences of approximately a 24 hour period, we see this comparison between the number of top attackers.

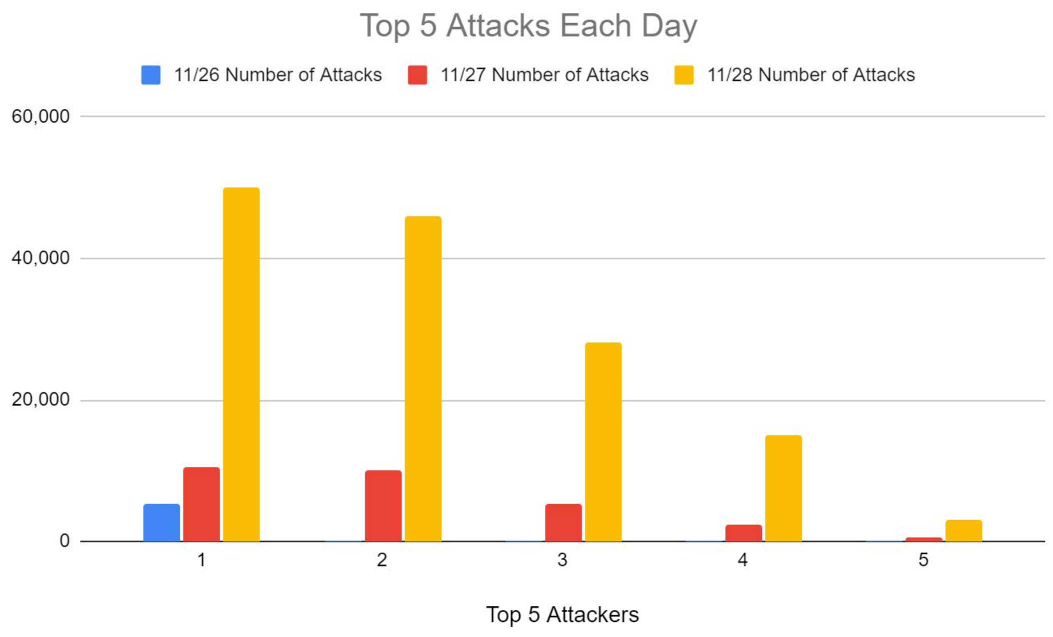


Figure 22: Comparing number of attacks from top 5 attackers between Nov26-28th

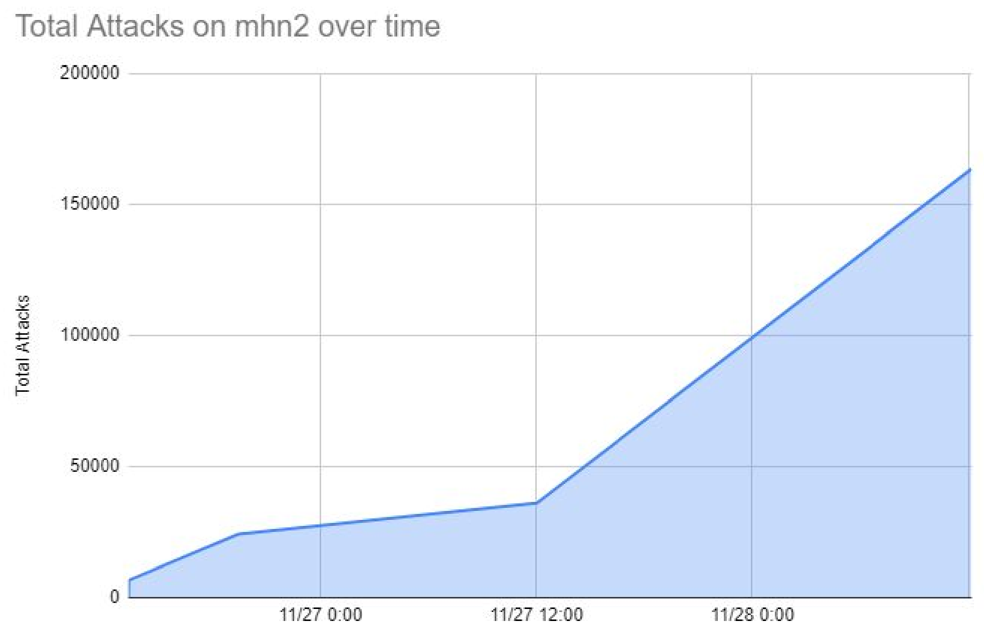


Figure 23: Number of attacks over time

The figure above demonstrates the increase of attacks after the set-up of the server.

Below we have a table of the attacker's IP addresses and the number of attacks conducted. From this table, we see some similar if not the same number of attacks for identical addresses from the attackers. This relation is seen through the highlighting of different cells of different times.

11/26 12:22:00 AM		11/26 7:27:00 PM		11/27 12:02:00 AM		11/28 12:11:00 AM	
IP Address	No. of Attacks	IP Address	No. of Attacks	IP Address	No. of Attacks	IP Address	No. of Attacks
47.52.144.177	5,224	85.116.124.124	10,048	222.211.70.93	10,445	222.211.70.93	50,066
212.73.68.131	114	47.52.144.177	5,224	85.116.124.124	10,048	160.124.255.14 8	46,039
31.14.84.1	113	14.170.192.111	2,289	47.52.144.177	5,224	167.114.223.21	28,032
80.191.208.185	112	103.143.206.243	118	14.170.192.111	2,289	192.186.59.82	15,044
117.1.58.146	111	31.145.150.194	114	211.20.149.211	630	117325.154.13 7	3,120

Table 1: Attackers and Number of Attacks

Since our MHN server got full from gathering data, we had to create another one to start collecting data again. Once the third Modern Honey Pot server was created, more consistent data was gathered. The figures below present the data gathered between November 30th and December 3rd.

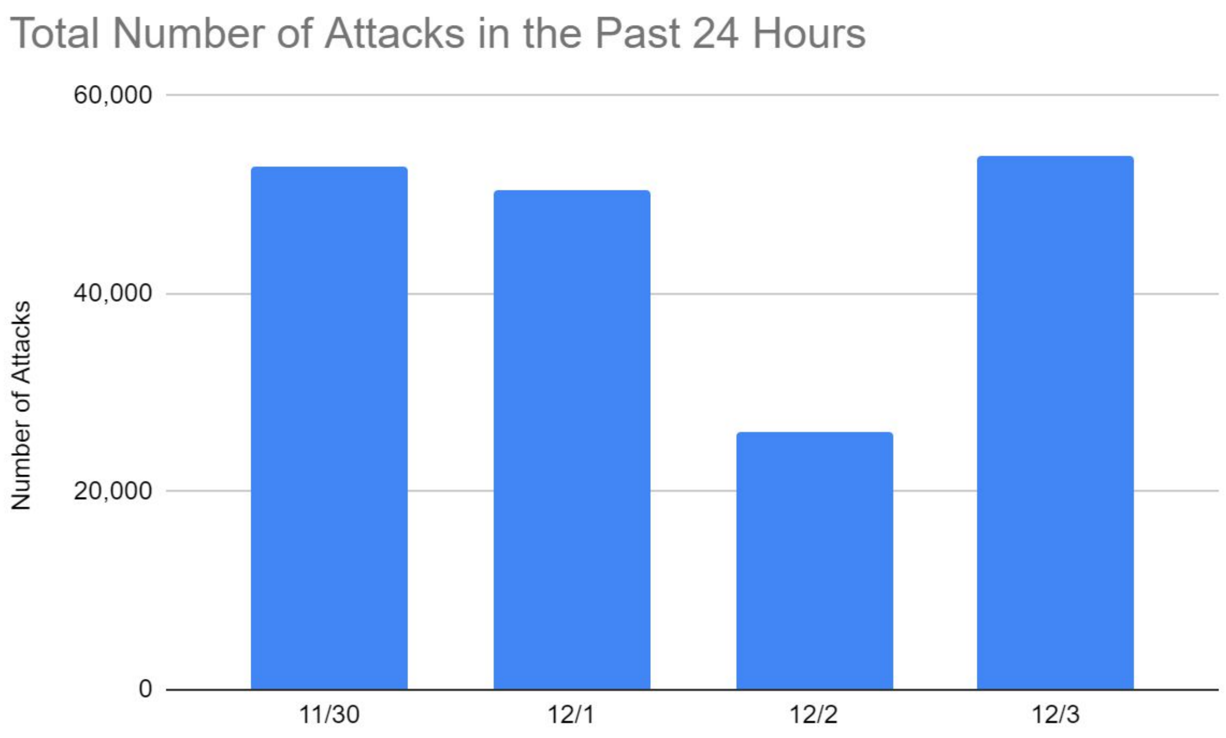


Figure 24: Third Server Total Attacks from Nov30th-Dec3rd

As expected, the number of attacks every twenty-four hours are roughly the same eliminating the outlier of December seconds. With the outlier, the average of attacks is 45,831. Without the outlier, the average number of attacks every twenty-four hours is 52,430 attacks. Taking this in consideration, it is no surprise that the servers were overloaded with data on how many attacks it recorded.

Top Attacked Ports			
Port Number	11/26 7:27:00 PM	11/27 12:02:00 AM	11/28 12:11:00 AM
1900		10,460	148,942
445	15,254	16,294	4,864
3389	202		3,647
23	423	481	1,557
80			358
1433	5,244	5,248	
8621	885	895	

Table 2: Chart of Top Attacked Ports

The above ports not only have specific protocols but also specific apps in which they are associated with. Once the third server was set up, the attacked ports were analyzed and some ports were similar and others were different. The ports highlighted in yellow are the attacked ports that did not show up in the results of the second mhn server. The ports highlighted in grey are ports that were not attacked in the results of the third server but were attacked in the previous server.

Top Attacked Ports				
Port Number	11/30	12/1	12/2	12/3
1900				
443		103		
445	39,382	32,540	16,780	26,988
3128	1,748	93		73
465	1,534			
3389			157	11,407
22		17,242	8,385	14,991
23				

8080	1,675		254	95
80	1,250	104	85	
1433				
8621				

Table 3: Top Attacked Ports for Server 3

The following table shows which ports are associated with which protocols and applications. When observing the different protocols and the assigned application, it can be inferred the intended attack attackers wishes to use.

Port number	Associated Protocol	Assigned App
1900	UDP	Microsoft for SSDP (Simple Service Discovery Protocol)
443	TCP	HTTPS (Apple Applications i.e. iTunes Store)
445	TCP	Microsoft-DS Active Directory, Windows shares (official)
3128	TCP	Active API Server Port
465	TCP	SMTP Mail over SSL
3389	TCP	Windows Remote Access (WRA)
22	TCP	Secure Shell (command line access)
23	TCP	Telnet
8080	TCP	HTTP port use for web traffic
80	TCP	Hyper Text Transfer Protocol (HTTP) - port used for web traffic
1433	TCP	Microsoft SQL Server.
8621	TCP	Unassigned

Table 4: Chart Port Number Information

As seen above, there are quite a few inferences that can be made. When observing port 80, it can be observed that the service is HTTP and thus susceptible to trojan and other web-based attacks such as worms and backdoors. The most common of these ports seem to be susceptible to trojan attacks.

Known errors:

The MHN server sometimes goes down and requires a reboot. After reboot this error occurs: unix:///var/run/supervisor.sock no such file

https://github.com/pwnlandia/mhn/issues/549

Fix:

$ sudo touch /var/run/supervisor.sock

$ sudo supervisord -c /etc/supervisor/supervisord.conf

$ sudo supervisorctl restart all

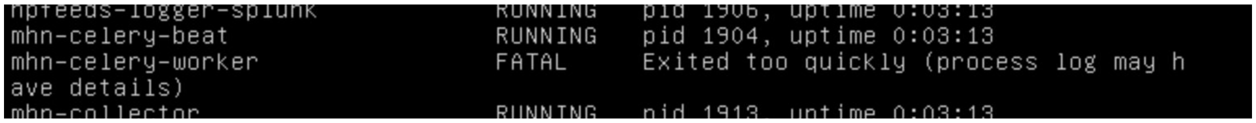
A script, post_reboot.sh, was created with the commands above to streamline the reboot process.

Currently some of the services that are provided by the MHN will go down randomly. In order to fix these problems the following site was visit for a solution:

https://github.com/pwnlandia/mhn/wiki

https://github.com/pwnlandia/mhn/wiki/MHN-Troubleshooting-Guide

One solution that was solved with this guide involved the following:



```
hpfeeds-logger-splunk            RUNNING   pid 1906, uptime 0:03:13
mhn-celery-beat                  RUNNING   pid 1904, uptime 0:03:13
mhn-celery-worker                FATAL     Exited too quickly (process log may h
ave details)
mhn-collector                    RUNNING   pid 1913, uptime 0:03:13
```

Figure 25: MHN-celery-worker being in fatal causes MHN to stop responding

The provided solution is as follows:

$ cd /var/log/mhn/

$ sudo chown www-data mhn.log

$ sudo supervisorctl start mhn-celery-worker

Future Work

For future suggestions based on the team's experience, it would be best to use Ubuntu 16.04 or 18.04. The team faced difficulties with using Ubuntu 20.10 since the MHN set up is not supported by python 3, which is the default python version for the newest Ubuntu.

Another suggestion is to use a bigger virtual server so that there could be longer periods of attack logs gathered. Although the honeypots are the targets, this caused the MHN to stop responding and gathering datas. Our second MHN server became full after a couple days and we had to remake another MHN to gather more data.

Next time our group decides to do this project, we would also want to extend the duration of data gathering. We have collected data for a whole week but it would be interesting to see patterns of these hackers over a long scale, such as months or years. This would allow us to have precise data and draw a better conclusion about these hackers.

However, this was hard for us because our MHN server will randomly shut down. When it shuts down, it stops collecting data until we manually restart it and run some commands. Therefore, we also would implement a bot/command that will periodically check if the server is up or not. If it is down, it will reboot and run the commands to set the server back up. This also caused inconsistency within our data collection because we are not available 24/7 to check the status of the servers and to restart it. This caused the MHN to be idle for multiple hours within our data collection span, hence may be the reason why some days have fewer attack counts than others. Due to the issues we faced with our virtual server, we did not have precise data collection schedule or timings. This caused the data gathered to be hard to scale and draw conclusions from.

Conclusion

Through the completion of this project, information about a multitude of different network based attacks has been recorded. The mechanism responsible for the capture of attacks' data has been the Modern Honeypot Network along with the numerous different honeypots. The Modern Honeypot Network server, MHN for short, was the main interface for viewing recorded attacks on the deployed honeypots. Over the course of the project three MHN servers were implemented.

The first MHN server can be considered a test server because we were still learning about the functionality and management of the MHN server. The two things that we learned from running this initial server involved honeypot's Ubuntu and python version compatibility along with troubleshooting fatal processes like the celery-worker or the hpfeeds-logger. The server accumulated just over 42,000 attacks on its single Dionaea honeypot over its four days of operation. About 39,500 of the attacks on the Dionaea honeypot targeted ports 445 and 1900. After the first four days of operation, processes started to constantly fail causing the MHN server to have short uptimes. The following day data displayed on the MHN server zeroed out. Not knowing what caused the issue we decided to take the MHN server down. Also during the setup we mistakenly deployed the Dionaea honeypot on the same server as the MHN.

The second server was short lived as the Dionaea honeypot's log filled it's 25GB of storage. The data displayed on the MHN server zeroed out again around the same time. We believed that filling the storage and the data not displaying correctly were related so we tried deleting the full honeypot. However, the data displayed didn't come back, so after attempting to troubleshoot the issue we ended up remaking the server.

The third MHN server was the most successful server as the uptime was fairly constant when compared to the prior two MHN servers. We deployed four honeypots, Dionaea, Amun, Cowrie and Snort, each on their own server. This was done in order to avoid any one of the honeypots from filling up their storage. Looking through the data recorded by the server it can be seen that there are a select few ports that have been subjected to a large quantity of attacks. Many of these attacks repetitively attack the same set ports thousands of times a day. Looking at the ports listed in Table 2 and Table 3 and corresponding functions on Table 4 it can be seen that these ports are related to remote access.

With the information recorded across all three MHN servers, precautions can be derived in order to protect highly targeted ports. The simplest method is blocking any port that might create a vulnerability. Blocking the port is a surefire method to protect a port, but it also can keep authorized users So another solution is using a firewall to block unauthorized users from accessing a specific port. In a commercial environment a honeypot network can be used to develop an intrusion detection system that can protect a single device or be scaled to protect an entire network.

References

Research

https://usa.kaspersky.com/resource-center/threats/what-is-a-honeypot

https://us.norton.com/internetsecurity-iot-what-is-a-honeypot.html

https://www.comparitech.com/net-admin/how-to-establish-a-honeypot-on-your-network/

https://www.imperva.com/learn/application-security/honeypot-honeynet/
https://www.speedguide.net/ports.php

Instructions, codes, troubleshoot

https://github.com/pwnlandia/mhn
https://www.youtube.com/watch?v=vUj9W0w7MdA

Summary of Activities and Contributions

Our group went into the research of how MHN works, what its purpose is, and how to create one. Each member researched and assisted in the set up of the modern honey pot server as well as the different honey pots themselves. Since the group is using an open-source MHN, we all went through the steps of setting up our own MHN server and honey pots themselves. After setting up, we obtained a deploy code that we pasted into the terminal of our honeypots. After pasting the deploy code, we observed the attacks coming in and collected the data to later analyze and discuss. As for the server, since it is hard to keep our device turned on 24/7 for a month we are renting multiple Ubuntu 16.04 servers on digitalocean. We want to keep our servers running so we can get more attacks and collect their data.

As mentioned above, everyone contributed to the research and setup of the MHN. In addition, we all individually initialized the MHN in case one of us was not able to set it up properly. Ryota and Ryan both created the MHN servers on digitalocean and made the honeypots, and reset the server when it was down. Ryan researched troubleshooting and solution when our MHN server shut down.

Once all the data was gathered, Danielle entered the data into tables on Google Sheets. Once the data was entered, she generated all the different charts and tables that are seen in the project report. During a meeting with the entire team, all members participated in the analysis of the data and the different ways the data could be observed. Danielle made more charts and tables to reflect the team analysis. The entire team contributed to the creation and completion of the project report. All the different sections of the report were split up among the team members to complete and then reviewed by the rest of the team members.