Building a Honeypot Server And Analyzing the Data it Collects

Jared Campbell
The University of Texas at Austin

David Zehden
The University of Texas at Austin

ABSTRACT

This project aims to build a honeypot server and analyze the data it collects. The goal of the honeypot server is to create vulnerable server on the open internet which we expect will be attacked by malicious actors. These attacks will be logged by the server, then we will analyze the data collected to find common trends in the attacks.

1. INTRODUCTION

A honeypot is a server that is made intentionally vulnerable in order to attract the attention of malicious actors. The server then logs any attempts by attackers to exploit and gain access to it. The goal is for researchers to be able to analyze common trends in the kinds of attack used by attackers and even possibly discover new kinds of attacks that have never been seen before. Another use case for honeypots in an enterprise environment is to slow down attackers by allowing them to attack the non-critical honeypot instead critical infrastructure.

There are many different tools that can be used to create a honeypot. For example, the honeypot could emulate a vulnerable web app or a vulnerable end-user machine. The configuration of the honeypot depends on the goals of its creators whether that be research or protection as described above. Our goal is to research these various tools and gather enough data to be able provide an analysis of current trends in attacks.

For our approach, we will configure a honeypot as a vulnerable server using various tools which we have found to emulate common vulnerabilities and log attack data. The key insight of our project is dependent on the data which collect as there a numerous outcomes depending on the types of attacks we receive.

2. MOTIVATION

Mairh, et al. discuss a variety of use-cases for honeypots, among them is the use of a honeypot in conjunction with an Intruder Detection System (IDS). The authors note that an IDS typically uses misuse and anomaly detection [1]. We hope to both implement a system similar to the one described by Mairh, et al. and perform an analysis of the data we collect from the system in accordance with the type of anomaly detection described.

Additionally, Chuvakin was able to classify individual types of attacks on his system [2]. Such a level of analysis is impressive, and we hope to achieve similar data collection, albeit in a lesser volume due to the significantly shorter time frame. Also, our analysis may be more focused on traffic metrics rather than the sort of in depth log analysis done by Chuvakin. We are more inclined towards determining if and when an attack has occurred rather than what specific attacks occurred.

3. OUR ARCHITECTURE

We have set up a basic Amazon Web Services server. The server is an EC2 t2.micro instance. We are using an Ubuntu Server 18 image as our operating system.

To create the functionality of the honeypot, we have found several tools which we will configure and test. Dionaea captures malware and can simulate certain individual vulnerabilities. Cowrie is an SSH honeypot which logs brute force attacks and shell interaction. In order to manage these honeypot technologies we will use Modern Honeypot Network (MHN). MHN provides a single platform for managing honeypots and offers several useful analytics tools for assessing high level attack metrics as well as management tools to easily modify, add, and remove honeypot instances.

For data analysis we will mainly use Python to graph trends. Additionally, MHN provides some nice data visualizations of attack data it receives from connected honeypots. Tanner is another tool we can test which analyzes data specifically from SNARE, mentioned above.

4. SETUP AND CONFIGURATION

We have set up Dionaea on our AWS server and it is successfully collecting logs. Some further configuration may be required in order to ensure that we are logging at an appropriate level. We also plan to install and configure more modules for Dionaea including some for logging and some for adding more vulnerabilities. We also have Cowrie running on our AWS server. Modern Honeypot Network is running and provides up to date data collected by Dionaea and Cowrie, which already provides some high level insights. We may add additional sensors to MHN later on.

We have also configured the AWS server's firewall and security groups. Currently we are allowing all inbound net-

work traffic on all ports and denying all outbound traffic. SSH access is restricted to our own SSH keys. Password logins are denied and root login is also denied. So far we have logged numerous attempts from scanners to log on to our service, and Dionaea has already captured a few malware binaries.

5. RESULTS

Due to some technical issues, we had to reset our EC2 instance, thus the findings presented are those collected following the reset. As of the writing of this paper, we have received over 1,300 attacks on our honeypots. Modern Honeypot Network provides some useful high level statistics about our honeypots some of which may be seen in tables 1 and 2. Another interesting observation is that the vast majority of our data came from Snort, as over 1,200 attacks originated on our Snort honeypot.

We also found that 1433 was attacked significantly than any other port. This is likely due to port 1433's use as the default port for many SQL servers.

IP Address (truncated)	Country	Number of attacks
45.136	Germany	21
80.82.7	Netherlands	18
83.97.2	Romania	17
89.248	Netherlands	17
81.22.4	Russia	16

Table 1: Top Five Individual Detected IP Addresses

Attacked Port	Number of Attacks	Common Port Use
1433	195	SQL Server
22	70	SSH
5060	68	Clear Text SIP, VoIP
445	18	Server Message Block
8545	17	Remote Procedure Call interface of Ethereum clients

Table 2: Top Five Attacked Ports

Using the PyGeoIpMap library, we were also able to plot the approximate locations of attacker IP addresses (Figure 1). Additionally, we plotted the top 20 countries by attacker, as seen in Figure 2. While IP addresses from all around the world were detected, the majority came from the United States, the Netherlands, China, Russia, and Germany.

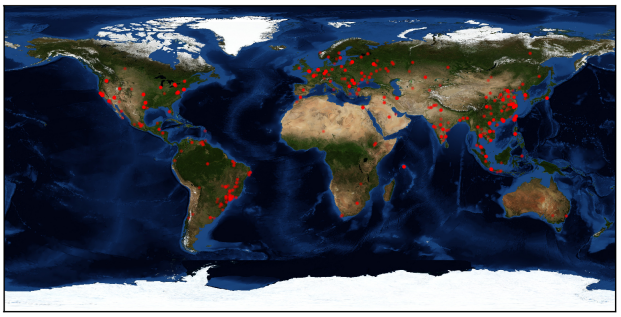

Figure 1: Approximate Locations of Attacker IP Addresses.

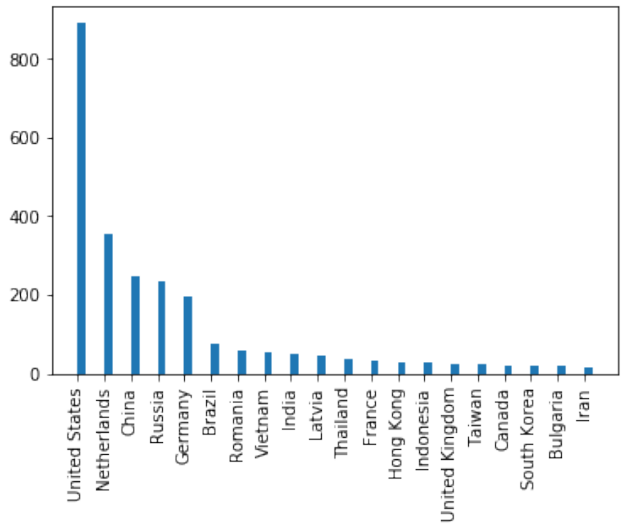

Figure 2: Top Countries by Attacker.

6. CONSIDERATIONS

Since we are using AWS to host our honeypot, we must consider the configuration of the AWS instance itself. In particular, we will consider AWS firewall rules and security groups to allow all incoming connections and deny all outgoing connections. We must also consider that AWS might shutdown our server if they detect it is under attack. In that case, we will have a backup of the server prepared and it will be migrated to a private server which we control. Finally, if our honeypot does not collect enough attack data in the time period during which it is active, we will attack the server ourselves to demonstrate its capabilities.

If time permits, we have further plans to add more AWS honeypot servers and network them together with our original instance to hopefully see how an attack might traverse the network. We can also implement an intrusion detection system such as Snortif we have enough time.

7. RELATED WORK

Steve Gathof wrote an excellent article on the setup of a honeypot on an AWS EC2 instance [3]. While his work will likely be a useful reference for our architecture, it is not explicitly related to the analysis that we will be performing.

Additionally, Polyakov et al. explored both the architecture and several specific functions that a honeypot should fulfill [4]. We will try to implement the functions mentioned in the Polyakov's article, in particular, creating variable conditions for accessing the system in order to possibly filter out low level intruders.

8. CONCLUSIONS

We hope to construct and analyze a honeypot server. We intend to perform an analysis of our findings in order to determine trends regarding the traffic our server receives. In particular, we hope to detect and examine anomalies on our system. We do expect to receive a some amount of traffic from automated sweepers and hope to attract more interesting attackers as well.

9. REFERENCES

[1] Mairh, Abhishek & Barik, Debabrat & Verma, Kanchan & Jena, Debasish. (2011). Honeypot in network security: A survey. ACM International Conference Proceeding Series. 600-605. 10.1145/1947940.1948065.

[2] Chuvakin, A. (2003). âĂIJHoneynets: High Value Security DataâĂİ: Analysis of real attacks launched at a honeypot. Network Security, 2003(8), 11-15.

[3] Gathof, S. (2018). Deploying a Honeypot on AWS [Blog post]. Retrieved from https://medium.com/@sudojune/deploying-a-honeypot-on-aws-5bb414753f32

[4] Polyakov, V. V., & Lapin, S. A. (2018, October). Architecture of the Honeypot System for Studying Targeted Attacks. In 2018 XIV International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering (APEIE) (pp. 202-205). IEEE.