Honeypot Configuration and Data Analysis

Jared Campbell & 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 a 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. Our honeypot will collect data on both an attacking machine and the attacks it directs at the honeypot. Our honeypot server is hosted on an Amazon Web Services (AWS) virtual server instance. The network for the server has been configured to allow all incoming traffic on monitored ports we expect to be targeted (such as SSH). We then analyzed the collected data using Python to find trends in attacker IP addresses, attacker operating systems, attacked ports, and timing of attacks. The honey pot was successful in collecting a large amount of attack data, averaging over 1000 unique data points per day. Our research has determined that a honeypot server is an effective tool for monitoring potential attacks on a network. We have also found honeypots to be highly configurable which allows for the collection of data specific to an organization's needs.

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 attacks used by malicious actors and even possibly discover new kinds of attacks that have never been seen before.

While many different types of honeypot tools exist, we used Modern Honeypot Network (MHN) as the backbone of our project. MHN provides a nice interface for adding particular sensors and viewing data collected over a period of time. As seen in Figure 1.

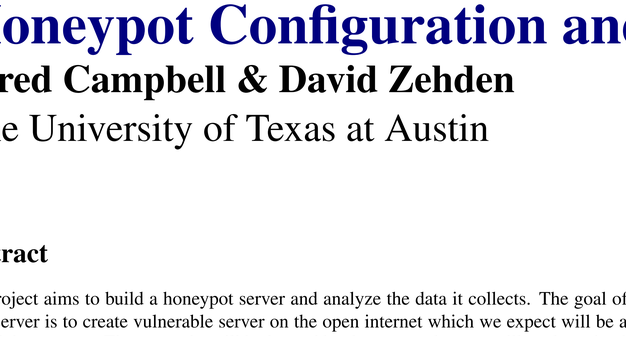

Figure 1: The Modern Honeypot Network Dashboard

Main Objectives

1. Setup Amazon Web Services EC2 instance
2. Explore Honeypot options
3. Deploy Honeypot of choice on AWS
4. Configure Honeypot as needed
5. Collect data over time
6. Explore collected data to determine trends

Setup

The three sensors used by the honeypot, Dionaea, Cowrie, and Snort, were each initially individually configured through MHN. We then manually configured the logging levels of each sensors to ensure relevant data was logged separately from error and debugging information so that it could be parsed by our Python analysis scripts.

We have also configured the AWS server's firewall and security groups. We are allowing all inbound network traffic on all ports monitored by the honeypot (SSH, FTP, HTTP(S), etc.) and denying all outbound traffic. SSH access is restricted to our own SSH keys, and the SSH port has been changed from the default of 22 to allow Cowrie to monitor attacks on port 22. Password logins are denied and root login is also denied.

For data analysis, we used the `mongoexport` command line tool to export data collected by MHN to JSON files. We used `SCP` to securely copy these files onto our local machines. Then we used Jupyter notebooks to explore data. Primarily, the `matplotlib`, `numpy`, and `pandas` libraries were used to aid the development of our findings. Additionally, we modified MHN to include a new webpage for particularly interesting findings. Such findings were generated from a daily `cron` job we set up on the server to run scripts that we developed locally.

Results

Our honeypot made over 20,000 detections. We received traffic from all over the world, as you can see in Figure 2. In order to generate this map, we took all the IPs we detected and used the PyGeoIpMap library.

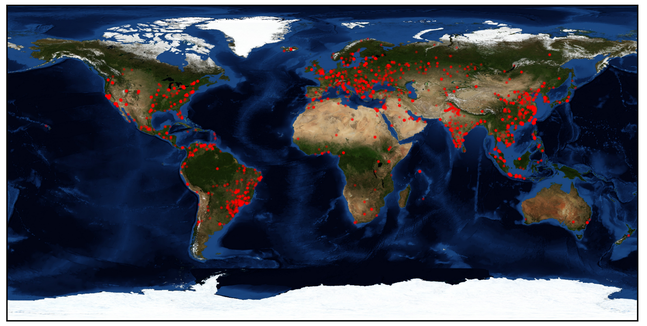

Figure 2: Approximate Locations of Detected IP Addresses

Additionally, we tried to break down the frequency with which we detected countries, as can be seen in Figure 3.

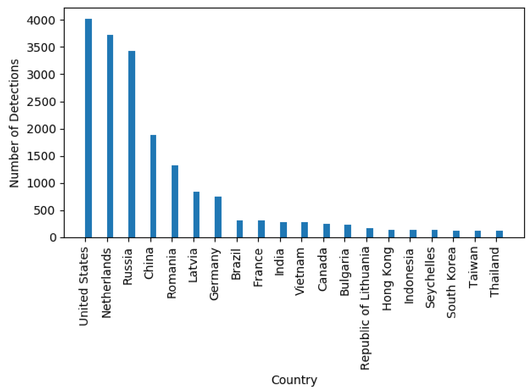

Figure 3: Breakdown of Detections by Country

We also broke down our detections by time of day, as seen in Figure 4.

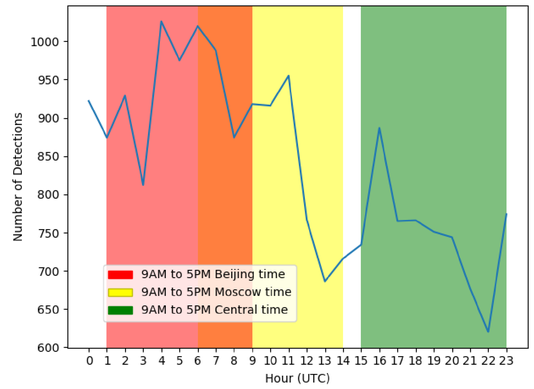

Figure 4: Breakdown of Detections by Time of Day (UTC)

Finally, we observed the types of operating systems that our "attackers" were using.

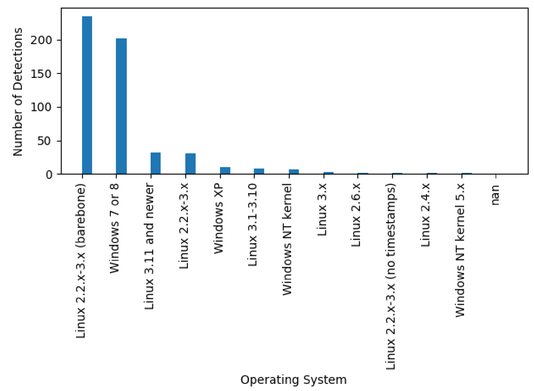

Figure 5: Breakdown of Detected Operating Systems

Conclusion

In order to research the effectiveness of honeypots, we implemented and configured our own honeypot on an AWS server. Our honeypot network collected even more data than we expected over the course of a single week. We have analyzed the data collected by our honeypot network to find trends in the origin of attacks, the hardware used for attacks, which services are most heavily targeted, and more.

We have also seen that different honeypot solutions offer both various types of data and different quantities of data. From our findings, we could see that Snort was was able to capture a massive volume of information, but it was data that was generally more surface level. In contrast, Dionaea could provide examples of malware binaries that attackers attempted to run. However, this happened very infrequently, so it ultimately did not provide a great deal of data.

Our research has demonstrated that honeypots are an effective tool for monitoring malicious activity on a network. Our research also shows that honeypots can collect large amounts of data on many different types of trends. The data collected by a honeypot can be tailored to suit any organizations specific needs, making honeypots an effective tool not only for research, but also for securing enterprise environments.