TED University/ CMPE491

Senior Project

Low-Level Design Report

Date

02-28-2022

Website of the Project:

https://ayse-nur-safak.github.io/HoneyTED/honeyted/index.html

GROUP MEMBERS	SUPERVISOR
AYŞE NUR ŞAFAK	EMIN KUĞU
KAYRA EGE ARDA	VENERA ANDANOVA
BARAN AKIN	TOLGA KURTULUŞ ÇAPIN
EBRU KILIÇ	

CONTENTS

1. INTRODUCTION

The rising vulnerabilities are one of the most regarding situations of the day. Therefore, every day, billions of attacks on small, medium, and large networks appear around the world. Some detection tools have been developed to protect networks by gathering information from previous malicious attacks. It is necessary to approach this problem with many different solutions because the habits of attackers, the way they attack, and their goals are changing every day. Therefore, a honeypot is one of these tools which traps attackers by using fake data that a hacker will attempt to steal from. In this way, it offers an option that we can analyze them, except that we have only tricked and caught them. In this project, we will design a low-interaction, SSH honeypot. It will be installed on a public server to investigate cyber-attack methods. This will gain the protectability and help the people who defend their system.

What is the purpose of the system?

The purpose of the system is to collect data about the behaviors of real-world attackers, using what is essentially a “trap” and by recording how they react when they fall into this trap, we can generate useful data about attack patterns and attack vectors. Later, by analyzing and processing this data we can predict future behaviors of malicious actors.

What is the design goal?

The goal of the design is such that to design a system that will be exploited “purposely” the trap and have a secure part of the system that holds the data for later analysis and usage. Ideally, we want the data to be secure and safe, and potentially offload the data to another device in between arbitrary intervals. Therefore, we know for certain that the data that is collected cannot be used for nefarious purposes by third parties.

The constraints of the project

The constraints of this project are to have a system to “be hacked” and for people to roam around in such a system so that the behaviors can be saved. Finding companies or hosting services to host such a project has proven difficult to say the least. Another constraint is the security of the data that is collected. We must segment the system carefully to have a part of the system that will be vulnerable, and the data storage part be safe and secure.

Professional and Ethical Responsibilities

Our professional responsibility lies in the security and the anonymity of the data that is collected. We must be certain that the data that is collected cannot be accessed by third parties, unless the law states otherwise. If the data, we keep is de-anonymized and is accessed by third parties we might be doing acts that are against data protection laws and this would both be non-professional and unethical. Never mind the illegality of such an act.

1.1. OBJECT DESIGN TRADE-OFFS

1.1.1. Usability vs Functionality

Since our currently ready honeypot is at the medium level, they require less functionality than high-level honeypots. It logs the attacker's steps during and after connecting to the machine. If we want to monitor this at the network level, it will further push the user in terms of usage. The reason for this is that the user can currently only view simple information in logs. The more functions are added, the more difficult it will be to use and understand.

1.1.2. Availability vs Maintainability

Since honeypots, by their nature, must be constantly under attack or taken, the system must be constantly turned on. The fact that the system is constantly open makes it difficult for maintainability. Steps to update, edit, or fix the system can keep the system turned off for a long time. This is a very big trade-off for an analysis system that should remain open all the time.

1.1.3. Security vs Cost

Honeypots are actually isolated and constantly monitored systems in themselves, but if they are not designed and coded in a secure way, they can also serve as a backdoor for attackers to infiltrate the actual system. In addition, a great deal of effort is also required to ensure the safety of the logs stored in the machine, to transfer and protect them with safe methods. All of this also returns in terms of time, labor and money. This shows how costly a process it is to ensure the security of the honeypot.

1.2. INTERFACE DOCUMENTATION GUIDELINES

This project is an implementation and research project. Therefore, we do not have additional codes that we have implemented into this system, hence we do not have any code class analysis at this moment. We will show the documentation with the diagram that we draw. In the diagram, lines show the relations between two objects. And under the object, it is written what that figure means.

1.3. ENGINEERING STANDARDS

Accepted engineering standards were used in this report. According to UML design principles, diagrams, class interfaces, subsystem compositions, hardware-software components depiction have been used to explain and visualize. Furthermore, we have incorporated IEEE standards while adding on to the project also.

1.4. DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

SSH: Secure Shell.

SDD: Software Design Document.

UI: User Interface

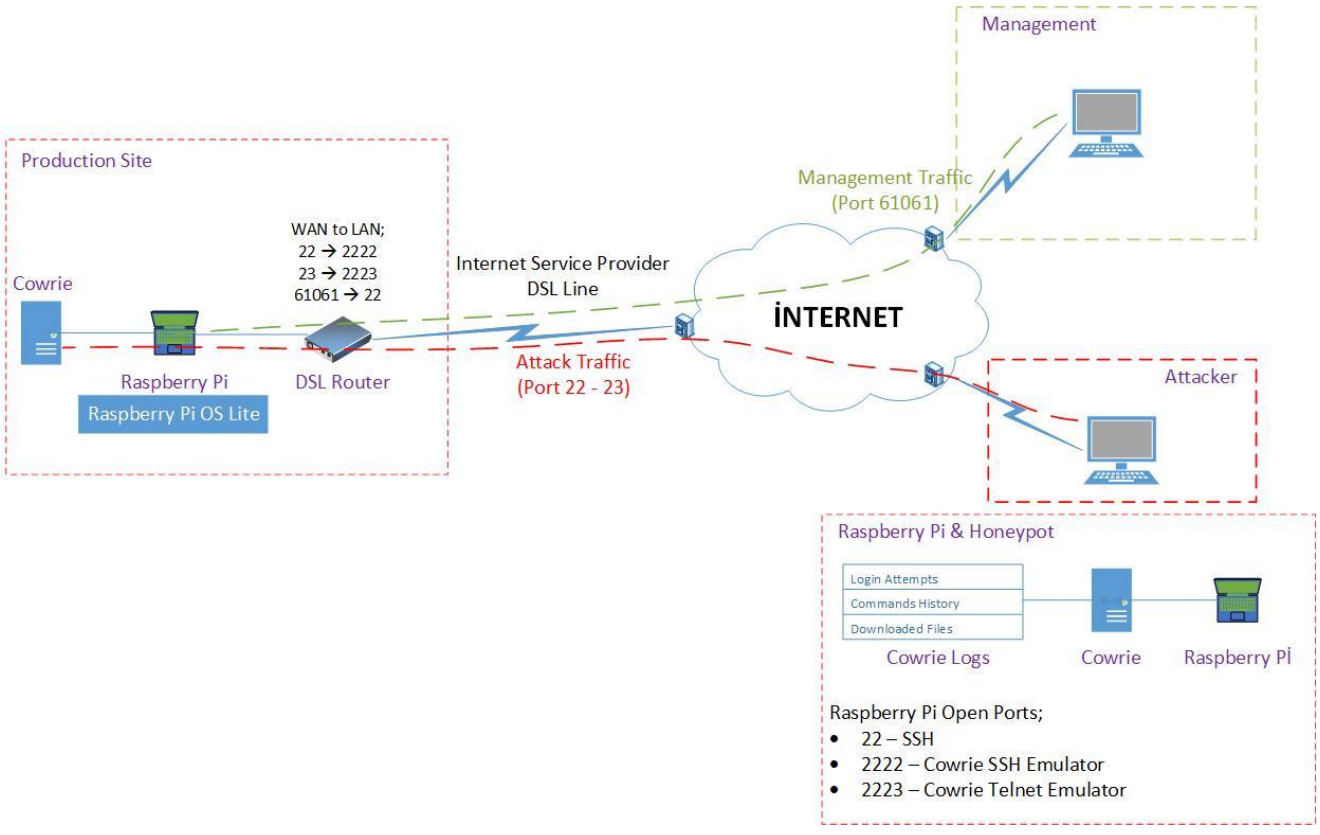
2. PACKAGES

Our system comprises glances, supervisord systemd and cowrie. Glances is used to track resource usage systemd is used to start these services automatically if the machine ever goes down supervisord is used to control and monitor the processes that are already running and our main application is the honeypot which is cowrie. We plan to add a static website as a bait application, our hopes are that when the hackers scan the machine that the static website is on they will see a vulnerability in the system which is the honeypot. These applications are all running on an server, our project doesn't require any client, even at the data collection action, we theoretically would be able to go next to the server, plug a cable and take the data off it, our usage of putty and connecting to the server using another port is not technically required nor it requires an client application even if we connect to the server, it is just a means of remotely accessing the server.

As a Web server, Apache is in charge of taking directory (HTTP) requests from Web users and delivering the requested data in the form of files and Web pages. We use it for operating our static website.

GPT-3, or the third generation Generative Pre-trained Transformer, is a neural network machine learning model that can produce any type of text from internet data. It was created by OpenAI and uses only a tiny quantity of text as input to generate large volumes of relevant and sophisticated machine-generated material.

Kibana is exploration tool and data visualization. We will organize our output. It used for log and analytics, monitoring and visualize the data. We will make our output more readable.

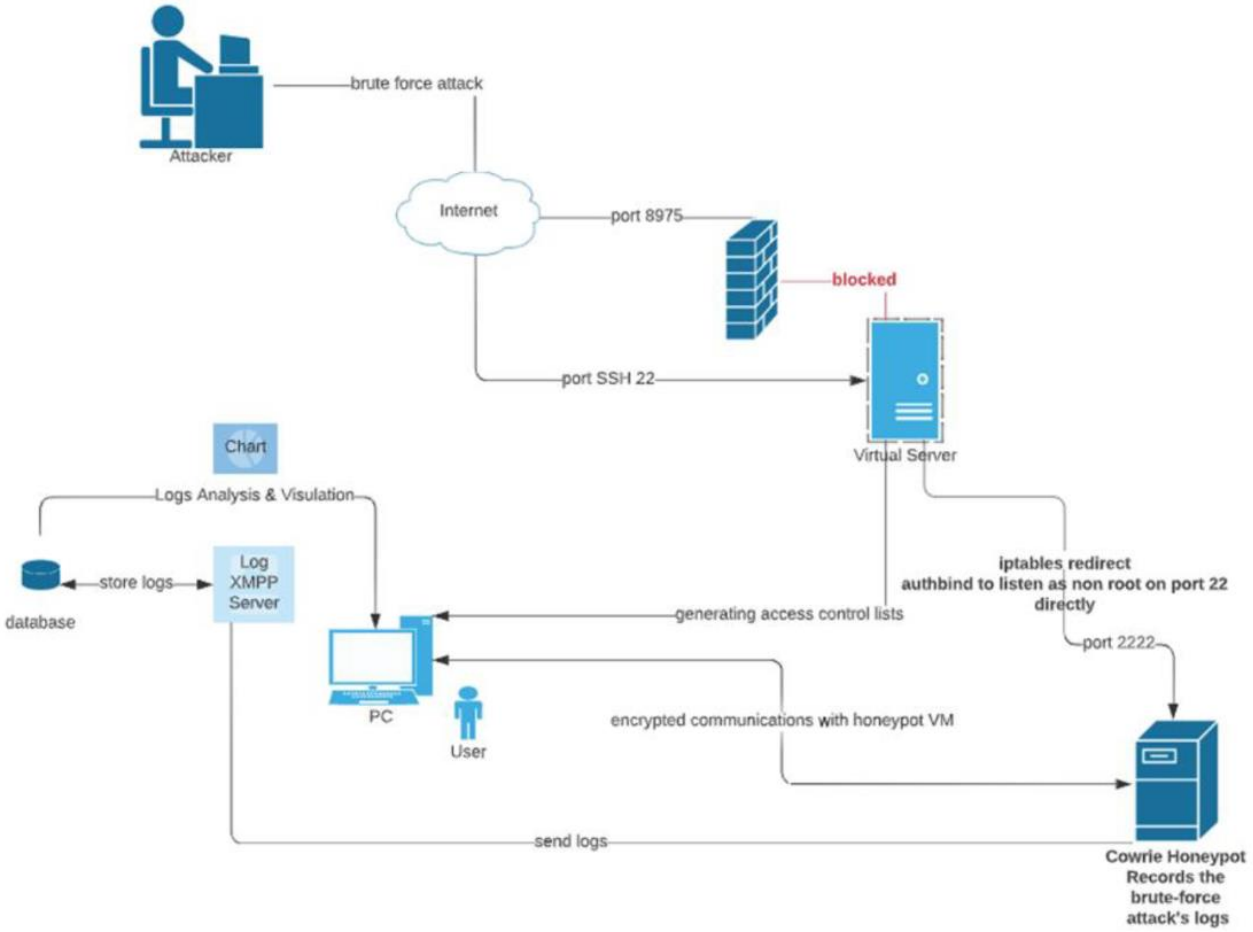


3. CLASS INTERFACES

Because this is an implementation and research-based project we do not have codes that we have implemented to this system, for that reason we at this time do not have any code class analysis.

On the other hand, we implement a static website to mask the honeypot to the intruders as a hosted website server, and a machine learning parser for the logs that will be generated, the machine learning will be based on gpt-3 since we do not know how to properly train a machine learning, yet we cannot give detailed information properly at this moment.

Therefore, with the masking of the honeypot, we believe that masking will generate more authentic data. by the increase in the quality and the quantity of the data, and the proposed development of the machine learning parser we hope that we will be able to generate useful results that will be used to predict future attack patterns.



4. GLOSSARY

In this section, there are explanations of the terms related to the project or system mentioned in the report.

Networks: A network is a technology that connects various devices, mainly for the purpose of data sharing.

Honeypot: Honeypots are trap servers that serve to collect information about attackers or users who have unauthorized access to information systems.

Malicious Attacks: They are attacks and software that are used to disrupt the functions of computers and mobile devices, collect critical information, provide access to special computer systems, and display unwanted ads.

Low-Interaction: Low-interaction honeypots use fewer resources; they collect basic information about the level, type, and where the threat is coming from.

SSH: SSH, or Secure Shell, is a remote management protocol that allows users to control and edit their servers over the Internet.

Attack Vectors: It is defined as the technique by which unauthorized access to a device or network can be provided by hackers for unfair purposes.

Offload the Data: Data offloading is the use of complementary network technologies to deliver targeted data for networks.

SDD: Software Design Document.

Vulnerable: vulnerability; errors that make a system or application vulnerable to cyber-attacks are called.

Anonymized: This means that data cannot be associated with an identified or identifiable natural person, even if it is matched with other data.

De-anonymized: The reverse process of anonymization.

Anonymity: It is the unidentified, unreachable, untraceable, and unknowable identity of a person or object.

Cowrie: Cowrie is a medium interaction SSH and Telnet honeypot library.

Python: The high-level language. The version 3.7+ will be used.

Glances: Glance is a system monitoring tool written in Python that uses the

psutil library to grab information from the system.

Supervisord: Users can be able to monitor and control a number of processes

on UNIX-like operating systems by using supervisord.

Systemd: Systemd is an init operation. The systemd works from the first time it is turned on to the last time it is turned on. In this way, we can run our honeypot and the necessary packages as soon as the system opens.

5. REFERENCES

1. [1] S. Russell and P. Norvig, Virtual honeypots: from botnet tracking to intrusion detection, 3rd ed. Addison-Wesley Professional, 2007.
2. ACM Code of Ethics and Professional Conduct.
3. The Software Engineering Code of Ethics, IEEE Computer Society.
4. IEEE Code of Ethics.
5. Computer and Information Ethics, Stanford Encyclopedia of Philosophy
6. https://github.com/cowrie/cowrie
7. https://cowrie.readthedocs.io/en/latest/index.html