Homework 5
CS 468: Network Security

November 14, 2023

Getting Started

The homework assignment is due on **December 1, 2023 at 11:59pm local time**. For this assignment, you are required to write your solution in Python programming language, version 3.5 or later. The goal of this assignment is to write a honeypot. To complete this assignment, you are allowed to use standard Python libraries, if you decide to use any non standard libraries, please ask on Piazza before proceeding. Make sure you read through the complete handout before starting.

Task 1: Creating an SSH Honeypot (20 Points)

A honeypot is a decoy system that is created to detect and study the presence of cyber attackers on a network. The function of a honeypot is to simulate the behavior of a real system with vulnerabilities such that it becomes a potential target for attackers. Specifically, when a honeypot detects an attack, it sends fake responses to the attacker pretending to be a vulnerable system. There are many kinds of honeypots, each specialized for a specific application. For instance, a honeypot running as a fake database server, when detects an SQL intrusion attempt, returns a legitimate query response. Similarly, an email address that has no other purpose can act as a honeypot to capture all spam traffic. In this homework, your task is to write an honeypot server running an `ssh` service, which detects a brute force attack, communicates with the attacker's `ssh` client and provides it with a shell access to a file system with the ability to execute a few basic commands.

You are required to implement your honeypot server so that it should initiate operation using the following inline argument:

```
>> python honeypot.py -p [port]
```

- `-p`: Is the argument of the port your `ssh` server will bind to. While the default ssh port is 22, your server should be able to detects attempts on any port.

There are three main requirements of this homework.

1. Detecting a brute force attack: Your honeypot server should be able to keep track of `ssh` attempts for a specific user. If the number of attempts exceed 5, your honeypot should grant access to a shell. An `ssh` attempt from the client can be generated using the following command, given you have `ssh` installed on your system.

```
>> ssh -p port USER@honeypot_ip
```

In order to simplify things, for this homework, you can assume that the honeypot is running on localhost. In addition, a list of usersnames has been provided in `hw5.zip`, which you can use for generating `ssh` attempts. You can also assume that the attacker only attempts to brute force one user at a time.

Hint: The password of the user would be irrelevant in this case as the honeypot is only keeping track of the number of attempts.

2. Implementing SSH functionality: Once the attacker (client) has been granted access and a connection has been established, your honeypot should communicate with the client which replicates a normal `ssh` connection. For instance, on a successful attempt by an attacker for the username `john`, should be able to see a shell prompt on the client side which resembles the following:

```
john@honeypot:/$
```

Just like within a normal shell, the client has the ability to type in commands which are sent over to your honeypot server when they hit carriage return. In addition, your server should also be able to keep track of time of an open connection. If the client remains idle for 60 seconds, it should terminate the connection.

Hint: Note that the attacker will be using an actual `ssh` client software to communicate with the honeypot, so your server should be able to understand the actual protocol and respond accordingly.

3. Implementing a basic file system: Once you have `ssh` implemented, the final task is to create a fake file system and a few related commands. On successful access, your client will be in the root directory. Initially, the root directory should be empty. You are required to implement these following system commands to allow the client to populate and interact with the file system.

- `ls` : for listing **all** files in the current directory
- `echo ''XXXX'' > destination.txt` : creates a simple text file with that filename and the content as "XXXX"
- `cat source.txt`: prints the contents of the source file
- `cp source.txt destination.txt`: creates destination.txt and copies the content of source.txt

Note : If the source file doesn't exist, cat and cp should explicitly print a `file X not found` error. The commands should handle any filename that follows the format *.txt. If a different or no extension is provided, the system should return an "unknown file extension" error message.
The example below provides a detailed demonstration of how the client should be able interact with the shell. The text in blue are the commands typed in.

```
john@honeypot:/$ ls
john@honeypot:/$ echo ''this is a sample content'' > source.txt
john@honeypot:/$ ls
source.txt
john@honeypot:/$ cat source.txt
this is a sample content
john@honeypot:/$ cp source.txt destination.txt
john@honeypot:$ ls
source.txt destination.txt$
john@honeypot:/$ cat destination.txt
this is a sample content
john@honeypot:/$ cat foo.txt
File foo.txt not found
john@honeypot:/$ cat foo
Unknown file extension
```

Testing

Testing for the honeypot system should be simple. You can test everything on your local machine, by running the honeypot server in one terminal window and then running an `ssh` client in another one, which attempts to make connections with your honeypot server.

Submission

For this assignment you are required to upload the following files in an archive named as `hw5.zip` to Gradescope.

1. `honeypot.py:` Your main honeypot program.
2. `explanation.txt:` Describe in a paragraph about your general approach for this homework and the list of online resources that you referred to.

Good Luck!