Build IoT Honey Jar

1. **What should this project be.**

This project would involve setting up a Linux/Unix box designed to emulate an IoT device, which would be monitored to assess the techniques that hackers use in attempting to compromise such devices, and may also predict the next target hackers is going to attack. As hackers typically aim to avoid detection, such monitoring must be implemented in such a way that it would be difficult for hackers to erase the monitoring data in order to ensure meaningful data collection.

2. **Inspiration for Idea**

The Internet of Things is the term used to describe the network of physical devices (e.g. vehicles, appliances, etc) connected to networks in order to exchange data with each other and the wider Internet. Such devices possess embedded systems running the software necessary to enable the network connectivity and manage the device functions. These systems tend to be vulnerable to exploits due to a combination of existing security solutions being unsuitable for the lightweight devices, and a reliance on network security rather than securing the individual device. Observing attacker behavior can aid in developing techniques to secure such devices.

3. **Steps we need to achieve**

The core elements of a honeypot architecture are Data Capture, Data Control, and Data Collection. Capture is concerned with monitoring attacker activity, control with containing attacker activity, and collection is concerned with storing the data (http://spi.unob.cz/papers/2015/2015-19.pdf). Thus, the key to ensuring the project is successful is to capture and collect data in such a way that it is difficult for the attackers to eliminate the data. Because we intend to run honey pots within a virtualized environment, and we do not want to give the attacker direct access to the collector, we need to implement the collector within the environment hosting the virtual containers that contain the honeypot - the host environment has access to the containers and thus the binaries corresponding to the virtual honeypots. Once the environment is established, we will collect and analyze the data, looking for patterns of interest, as noted in more detail below.

More specifically, we are interested in deploying multiple **Telnet honeypots**, using https://github.com/Phype/telnet-iot-honeypot as a starting point.

An IoT device generally has one or more of Telnet, SSH, HTTP and CWMP protocols enabled. Telnet is the most attacked protocol, because it is unencrypted, which makes it easier to target. Although new and more secure versions of Telnet have been developed, they are not very popular. Despite the drawbacks, it is still used by a lot of IoT devices due to the fact its implementation is relatively simple.

Thus, we will generate a honeypot that emulates interactions of Telnet protocol and various IoT devices. We will analyze some major attacks via Telnet protocol, which means the honeypot must be made to work with every form of automated telnet session, which may try to infect the honeypot with malware. Then we will design an application which has a client/server architecture, with a client (the actual honeypot) accepting telnet connections and a server aggregating connection data and sample analysis.

What we will achieve:

1. Observing current IoT attacks
2. Analyzing IoT attacks
3. Understanding infected IoT devices.

In observing current IoT attacks step, we will know the general flow of Telnet based attacks for attackers or the major attacks that come via Telnet protocol, including the attackers attempt to login to our honeypot, or send commands over Telnet download and execute the binaries then conduct the intended malicious activities.

In regards to **logging attacker actions**, there are two possibilities:

1. If honeypot is in VM, then store log on host machine in a directory that is shared as a write-only volume with the VM.
2. Ship the logs off as soon as they are created to another machine (could be another VM).

In analyzing IoT attacks step, we will do:

1. Intrusion Pattern: Pattern of User ID/Password challenges
2. Infection: The telnet command sequences attackers use to infect.

In understanding infected IoT devices step, we will analyze different types of devices visit IoT Honeypot, categorize the devices types, and these vulnerable IoT devices' area distribution, etc. If we have time, we could also do SSH-based, HTTP-based IoT Honeypot, and compare the results.

Tentative Timeline:

First two weeks: This will be an experimental stage. We will run different off-the-shelf honeypots found on https://github.com/paralax/awesome-honeypot on Microsoft Azure and/or AWS and/or Google Cloud. All of these cloud platforms provide free trials for new subscribers. We would look at the data collected and see which honeypots are the most worthwhile to continue running. These would be the ones that are attacked the most often and log the most data. We would continue to run these honeypots for the remainder of the semester.

Last two weeks: This would be the data analysis stage. We will leverage research already done on honeypots to come up with threat models for the attackers. Types of analysis we might do include Markov chain modeling of attacker behavior inside the system (if attacker does A, he will most likely do B). Other interesting statistics include arrival patterns of attackers (e.g., determining whether arrivals can be modeled via a Poisson process), clustering of attackers (K-means clustering based on various attributes including the timestamp of attack, files accessed, and time spent in the honeypot).

4. Demo

The demo would consist of demonstrating the environment containing the emulated honey pots and the data collection process. We would then proceed with presenting our analysis of the data collected, including arrival patterns of attackers, attacker origin, various attack methods and targets, correlation between differing event types, etc.

5. Assessing Success

Success would be if we collect a meaningful set of data and produce an analysis that gives insight on the patterns, tools, behaviors of attackers.