An IoT Honeypot Device for Malware Forensics

Speakers

Jingyu YANG, Senior Security Researcher, Tencent

Fan DANG, Ph.D. Candidate, Tsinghua University

Authors

{Jingyu YANG, Jie LI, Chen GENG, Zhao LIU, Guize LIU, Jinsong MA} @Tencent

{Fan DANG, Yongfeng ZHANG, Prof. Zhenhua LI} @Tsinghua University

Abstract

Tencent Anti-Virus laboratory has been observing the boom of IoT device-oriented malware against a backdrop of surging IoT device deployment. The targets of cybercrimes have been gradually turned to IoT devices rather than traditional PC or mobile phones, and the out-of-date honeypot technology cannot provide sufficient evidence during the malware investigation. In response to the emerging challenges, Tencent has developed a brand-new kind of IoT honeypot for IoT malware forensics.

Comparing to the traditional honeypot technology, it is a high interaction honeypot (HIH) that provides more information for IoT malware investigation and forensics. Firstly, the bi-direction network traffic will be captured, which means that the recorded data contains not only the traffic aiming to attack the device, but also the traffic initialized by the infected device itself. Secondly, common network services are provided, including SSH, Telnet, HTTP, UPnP, and even video streaming. All the services contain dedicated remote code execution vulnerabilities. Once they are compromised, the exploits and malicious actions will be monitored and reported to the management center as digital forensics. Finally, a net flow proxy module can be optionally deployed on the front layer. Instead of deploying the device everywhere, the module will redirect and aggregate attacking flow to pre-set honeypots to increase the coverage of capturing the attacks globally. The honeypot cluster also benefits from this proxy module for scalability.

The speech starts with addressing the disadvantages of traditional honeypots under the IoT environment. Then the architecture and the implementation will be introduced, followed by the reasons why the IoT honeypot solution help solve the aforementioned challenges. At the end of the speech, a case study of a real IoT attack captured by the honeypot will be presented.

1. Introduction

Before we introduce the architecture of this honeypot, please let me give some general information about the honeypots and the IoT honeypots. The three major questions are: how we categorize the honeypots; what are the differences between traditional honeypots and IoT honeypots; and the challenges we may face in the IoT environment.

There are two broad categories of honeypots available, high interaction and low interaction. So why do we care about the interactions? In my opinion, interaction measures the activity that a honeypot allows the attacker. It sets up the capability and limitation of a honeypot. Under the certain interaction, what attackers can do and cannot do are already determined. As a result, the more interaction you allow, the more you can learn. And the more interaction you allow, the complexity and risk you have.

Here is the comparison of two typical interactions. Low Interaction Honeypots allow only limited interaction for an attacker or malware. All services offered by a Low Interaction Honeypots are emulated. Thus, Low Interaction Honeypots are not themselves vulnerable and will not become infected by the exploit attempted against the emulated vulnerability. High Interaction Honeypots make use of the actual vulnerable service or software. High-interaction honeypots are usually complex solutions as they involve real operating systems and applications. In High Interaction Honeypots, nothing is emulated everything is real. High Interaction Honeypots provide a far more detailed picture of how an attack or intrusion progresses or how a particular malware executes in real-time. Since there is no emulated service, High Interaction Honeypots helps in identifying unknown vulnerabilities. But High Interaction Honeypots are more prone to infections and High Interaction Honeypots increases the risk because attackers can use these real honeypots operating systems to attack and compromise production systems.

The next question that we concern is the difference between traditional honeypots and IoT honeypots. There are already lots of honeypot implementations. You may find hundreds of open source repos in GitHub. However they are mainly targeted to Internet services. The architectures of traditional honeypots are mainly x86 and x86-64. However, the architectures of IoT honeypots are heterogeneous. The typical architectures are ARM and MIPS, but there are also many other architectures like SPARC PowerPC.

The services that the devices usually provide are also different. The web servers usually provide remote management service and other web services like HTTP, FTP. While the IoT devices usually provide specific services like video streaming and other kinds of private protocols. As a result, the existed simulators of certain services may not apply to IoT honeypots.
Besides, the deployment is quite different. The traditional honeypot can be easily deployed in the cloud. Unfortunately, if we need various architectures, we have to set up physical devices locally, which is quite hard to deploy.
To summarize the challenges we face in the era of IoT, we believe the major challenges are caused by the two factors. The first challenge is that it is quite difficult to adapt to different architectures. If we want to build a LIH, we lack of knowledge of how to simulate a service on an IoT device. But if we want to build a HIH, using emulators is also quite hard because there are no existed solutions to emulate commercial IoT devices.

The other challenge is that if we want to deploy the real HIH, it is both expensive and difficult. Because none of the existing virtual machines are applicable.

Therefore, we need a brand new architecture to design and implement the IoT honeypot. Next, Jingyu Yang will introduce the architecture and implementation.

2. Architecture

It is my pleasure to introduce the architecture and the implementation of our honeypot. Firstly, I would like to explain the whole picture of the honeypot project. Then two interesting components will be introduced. In the end, a case study will be discussed to demonstrate how our honeypot system works for hunting a new kind of cyber-attack.

2.1 The Whole Picture

First of all, this is the architecture about the honeypot project. As we can see, there are three main layers: Access Layer, Storage Layer and View Layer.

Access Layer

The access layer responses for accepting the network attacks from all kinds of sources and then collecting the payloads and reporting the malicious actions to the storage layer.

In the access layer, we developed a load balancer component and several different kinds of honeypot agents as backend execution component, such as SSH and Telnet agents.

Data Layer

When the access layer receives network attacks, it will report to the middle layer. The storage layer responds for data storage, statistics and data mining. The reported data will be stored in the MongoDB and processed information will be provided as JSON format to the view layer.

In the future, we plan to open the data upload interface and share the information with security researchers to defeat cybercrimes. A publish and subscribe system named hpfeeds is adopted, since the hpfeeds protocol has been used widely among many kinds of honeypot projects.

View Layer

We also designed a web management system as view layer, which can provide both human readable information and digital forensics. In the view layer, the administrator will be able to install and uninstall honeypot agents. And special users can export digital forensics such as PCAP files, which will be very helpful for the further investigation.

In the next, I would like to introduce two interesting components in the access layer.

2.2 Load Balancer

We created a load balancer component for redirecting cyber-attacks from original access points to the backend execution component. But why?

Because this component will ensure that malware in the backend execution component will not bring any harmful impact to the other nodes in the same level of access points. Based on the design that we separate the access points and the backend execution component, we can focus on monitoring malicious actions instead of preventing malware escape.

Our clients demand us provide more information about attackers. They want to know not only about the IP of the attackers, but also who they are and exactly what they did after the intrusion.

However, the threat that attackers manipulated honeypots to attack other victims should be handled properly. Any client will not accept that their honeypots turn to springboards which help to penetrate their own business instead of defending the attacks.

During the development of the load balancer component, we faced a technical challenge, which is how to forward the real IP of the attacker instead of the one from the load balancer.

To solve this issue, the load balancer will also report TCP connection pair to the storage layer. At the storage layer, log collector will merge the two-relative information. In the end, the real IP address of the attacker will be found and be linked with the session ID which represents one intrusion attack.

2.3 Yet Another SSH Honeypot

Regarding the backend execution component, we will discuss the SSH honeypot, since SSH oriented attacks are major parts of the IoT threats.

Our design principle is to build a monitor system based on a real execution environment includes hardware and embedded Linux. We patched the source code of dropbear which is an open source SSH server and deployed the program on Raspberry Pi 3. Comparing with traditional SSH honeypot, such as cowrie and kippo, a monitor system instead of an emulation system will bring many advantages.

For example, our honeypot agents will provide not only about username and password, but also a command list that attackers typed in. And the commands will be executed directly in our honeypot instead of being emulated in a virtual environment. What' s more, network data and CPU usage information will be reported to the storage layer. This extra information will be helpful to investigate new kind of cybercrimes, such as crypto-currency mining and ransomware. All the reported information will be documented in the storage layer as digital forensics.

However, management for hardware based honeypots is not an easy job. We found two technical challenges.

One is about how to reinitialize after the previous intrusion, the filesystem will be modified after the intrusion.

We built a special Linux kernel from scratch with boot argument as root=/dev/ram. In this case, the kernel will mount root file system in the ram. Any modification to the filesystem will be lost after restart. Our SSH honeypot will restart if any command has been executed in a session or wait until timeout.

The other is about how to response hardware failure. We designed a power controller with a programmable relay and the heartbeat system will send a signal to make the device restart physically.

3. Case Study

In the next, I would like to introduce a new kind of SSH attack to demonstrate how our honeypot works.

Our case study is about pivoting attack, which is a file less and pure network attack. In the pivoting attack, once the attacker gets the correct credentials of the SSH server, he will turn the server to a proxy by setting dynamic port forwarding option. Then the attacker will launch the DoS attack program at his own side. Eventually, the network flow will be redirected to the target. At the victim' s view, the IP address of the attacker is from the proxy server. By this way, the real attacker will be covered.

During this year, a huge number of credentials were lost. And most of them were IoT devices. This kind of attack contains three main steps.

Firstly, the attacker use brute force to get the correct pair of username and password. Secondly, the attacker uses SSH command with dynamic port forwarding option to setup a proxy server. Finally, the attacker uses DoS program with the proxy configuration to attack the target.

In the session report data, we monitored a huge number of pivoting attack to a bitcoin website. And the website also announced that they were under a DDoS attack on twitter.

The pivoting attack is different with the traditional intrusion attack. Firstly, it is pure network attack, any malicious file will not be downloaded on the file system. Secondly, the real attacker will be hidden behind the SSH server, which has been turned into a proxy.

3.1 A Hardware Based Honeypot

Many people asked me why you choose to develop a hardware based honeypot instead of software based solution?

Let me recall the benefits that hardware based honeypot brings and try to answer this question.

Firstly, as this case study demonstrated, our honeypot can provide accurate information about real attackers and victims. For example, traditional SSH honeypot, such as cowrie will not be able to capture new kind of cybercrimes, such as pivoting attack. Because, technically, they are SSH protocol emulator, and the dynamic port forwarding has not been implemented.

Secondly, anti-honeypot technology has emerged to help malware detects virtual environment, but our honeypot is based on real device and operating system, which means it immunes anti-honeypot technology naturally.

Finally, IoT devices are based on a variety of CPUs, such as ARM and MIPS. Most IoT firmware will not be loaded directly in emulators. However, these IoT devices can be turned into honeypots with slightly modification.

5. Conclusion

Although various honeypots have been widely deployed, they cannot be utilized directly. Because there are several challenges we have discussed the in the introduction. As for IoT devices, simulators and low interaction honeypots may be used. Unfortunately, the data collected by the LIH is limited. They are not sufficient for forensics. Therefore, if we want capture the full trace of attackers' behaviors, and if we want to collect as much information as we need for forensics, we need a HIH with a brand-new design.

In this talk, we give a new design of honeypot, which is targeted as IoT devices. To achieve a better performance, we employ the Raspberry Pi to be the honeypot. We use the new system architecture to redirect the data stream to our customized raspberry pi. As a result, we can simply enlarge the scalability to deploy the high interaction honeypot with collecting enough data as proofs.

We believe that this brand-new architecture and design provide every detail of an attack, which is quite good at identifying the malware in the era of IoT.

6. References

1. https://github.com/tencent/habomalhunter
2. https://github.com/threatstream/mhn
3. https://github.com/threatstream/hpfeeds
4. https://www.usenix.org/system/files/conference/woot15/woot15-paper-pa.pdf

5. https://researchcenter.paloaltonetworks.com/2017/07/palo-alto-networks-showcase-iot-honeypot-research-black-hat-2017/
6. http://blog.malwaremustdie.org/2017/02/mmd-0062-2017-ssh-direct-tcp-forward-attack.html