

Brief Industry Paper: Catching IoT Malware in the Wild Using HoneyIoT

Yiwen Xu*, Yu Jiang* , Lu Yu† and Juan Li‡

KLISS, BNRist, School of Software, Tsinghua University*
National University of Defense Technology, Changsha, China†
China Central Depository & Clearing Co.,Ltd., Beijing, China‡

Abstract—Constantly increasing botnets powered by vulnerable IoT devices perform record-breaking DDoS attacks to critical infrastructures. Therefore, it is imperative to find vulnerabilities in IoT devices ahead of attackers. In this paper, we first present a systematic analysis on various kinds of IoT malware to further explore the challenges of IoT Honeypot design. We then propose HoneyIoT, a scalable IoT Honeypot framework, which aims at attracting IoT attacks and recording malicious behaviors with configurable vulnerabilities and firmware support. During a 7-day real-world industrial experiment, HoneyIoT observed over 12,500 malicious connections and 3,423 distinct login attempts.

Index Terms—computer security, IoT malware, IoT Honeypot

I. INTRODUCTION

Internet of Things (IoT) is one of the most frequently-used technologies in our daily life. As we witness the growth in IoT devices, the alarming number of its vulnerabilities also reminds us that IoT malware is still at large and continues evolving. On October 21st, 2016, Dyn, a DNS provider acquired by Oracle Corporation, suffered from an unprecedented Distributed Denial of Service (DDoS) attack launched by Mirai botnet [1]. The attack exceeded 1 Tbps and cut access of millions of users to numerous websites.

To prevent those systems from attacks, many techniques are developed. For example, Honeypot is regarded as one of the most proactive methods to capture hidden fire-new malware. Honeypot disguises itself as a real system with applications and data, fooling malware into thinking that it is a valid target and then intruding into it. After in-depth track and analysis of the invaded malware, developers and maintainers could be forewarned of possible threats.

However, this plausible solution does not perform well in the IoT field. Actually, manifold architectures, communication protocols and device characteristics pose challenges to establish a dedicated IoT Honeypot. On the one hand, as IoT malware targets a wider range of IoT devices, it needs to have various attack modules for different architectures. For instance, Mirai, a notoriously disruptive IoT malware, is compiled to run on at least six processor architectures. Thus, back-end architecture support for IoT Honeypot is supposed to be strong enough to cope with the variety. On the other hand, vulnerabilities in IoT are related to the version and vendor of its firmware. In order to find out that specific vulnerable targets, tricky IoT malware employs distinct protocols and commands to carry out a deep pre-attack inspection of devices, which are probably IoT Honeypots. Without well-built frontend and

firmware support, the poor response may take a toll on the availability of IoT Honeypot.

To bridge the gap, this work thus conducts a systematic analysis on IoT malware to pinpoint the heterogeneous features of IoT devices it usually utilized during the invasion or attack. From analysis results, we identify challenges of IoT Honeypot and emphasize the importance of its front-end interaction capability and the back-end implementation. We then present HoneyIoT, a scalable IoT Honeypot framework equipped with configurable vulnerabilities and firmware support. During a 7-day industrial experiment, HoneyIoT observed over 12,500 malicious connections and 3,423 distinct login attempts.

II. IOT MALWARE AND IOT HONEYPOT

A. Investigation on IoT Malware

Table I presents the related information of some prevalent IoT malware. After studying and cross-validating threat reports, measurement papers, relevant blog posts and code in Github repositories, we describe some malware in detail, concentrating on the invasion and attack components.

1) *Invasion Methods*: Invasion can help IoT malware get a foot in the door when it comes to further destructive attacks. IoT malware usually employs two main methods to invade targets. One straightforward way is *brute-force password cracking*. On account of weak password settings or lack of modification to default passwords, IoT malware can attempt to gain access to an account with all possible combinations of the most common usernames and passwords in a cracking dictionary. Password cracking is so easy-to-implement and effective that it becomes the first choice of IoT malware.

However, the race among IoT malware is aggravated by decreasing uninfected IoT devices with weak passwords. Therefore, *exploit* gives some wise IoT malware such as Gafgyts [2], Hajime [3], etc., a decided advantage over their opponents. Specifically, these kinds of malware leverage novel or 0-day exploits to avoid sharing devices, which can be compromised with fewer technical skills, and increase the chance of infecting more vulnerable devices than those that do not. In order to spread more widely than the previous malware, for example, the new variant of Gafgyt targets Huawei and Asus routers by exploiting completely new vulnerabilities—CVE-2017-172151 and CVE-2018-15887.2—and removes 48 kinds of rival malware. It also performs a password cracking attack using several common login credentials. In addition, most of the vulnerabilities are related to specific devices. Therefore, when invading by exploits, IoT malware will conduct in-depth

 Yu Jiang is corresponding author.

TABLE I
THE ATTACK AND INVASION METHOD OF IOT MALWARE

Attack	IoT Malware	Invasion Method		Description/Particularity
		Crack Pw.	Exploit Vuln.	
DDoS	Mirai	✓		>DDoS Type:GRE/TCP/STOMP/DNS/UDP
	Psybot	✓		>Target mipsel modems and routers >Reside in RAM
	Persirai		✓	>Target webcam >Spread by SSDP vulnerability in LAN
	Gafgyts	✓	✓	>Remove rival malware >Terminate important system services
	Hoaxcalls		✓	>Using 12 IoT vulnerability exploits >DDoS type: UDP/DNS/HEX
	IoTReaper		✓	>Using 9 IoT vulnerability exploits >Embed Lua scripts for attacks
	Mozi	✓	✓	>P2P Communication (Multi C&C) >Target unpatched routers and DVRs >Use encryption algorithms to hide
	Hajime	✓	✓	>Target ISPs and MSSPs >Download other malware for help (eg:Brickerbot) >Remove firewall rules >Stay under the detection radar
Mining	LiquorBot	✓	✓	>Mine for Monero (XMR) cryptocurrency
	Darloz		✓	>Install epminer(a mining program) >Exploit a vulnerability in PHP >Mine for Mincoins or Dogecoins
Phlashing	BrickerBot	✓		>Corrupt MMC and MTD storage >Delete all files >Disconnect from the Internet >Use Tor network to hide its location

reconnaissance to determine whether the device, currently under attack, is the target containing specific vulnerabilities.

2) *Attack Methods*: The attack methods of IoT malware are diverse, including DDoS, Cryptocurrency Mining and Phlashing (devices' firmware damaging). The DDoS attack is the primary purpose of most IoT malware. To recruit new devices, the malware is intended to spread itself across the Internet by looking for vulnerabilities of exposed devices. Through a network of remotely controlled, hacked devices, they form what is known as a DDoS "botnet" or network of bots. These are used to flood targeted websites, servers, and networks with more data than they can accommodate. Table I lists eight types of popular IoT malware perpetrating DDoS attacks. It is conceivable that the IoT's large population and vast distribution pave the way for the considerable DDoS attack traffic.

Mozi [4], for instance, first surfaced online in late 2019, forcing routers and DVRs that are either unpatched or have weak passwords to join the botnet army. According to IBM, the Mozi botnet has surprisingly swollen in size, accounting for 90 percent of the observed IoT network traffic between October 2019 and June 2020. It evolved from the source code of three notorious malware families—Mirai, Gafgyt and IoTReaper. Mozi has blended them together to form a malicious peer-to-peer (P2P) botnet capable of DDoS attacks, traffic obfuscation and command or payload execution. It depends on a custom extended Distributed Hash Table (DHT) protocol, which is used by bots and P2P platforms to look up and store contact information, to establish its P2P botnet network without center servers. Besides, ECDSA384 and the XOR algorithm offer robustness and security to the nodes and

TABLE II
CRITICAL PROPERTIES OF IOT HONEYPOTS

HoneyPot	Interaction Capability	Backend Implementation	Real/Virtual	Open Source	Scalability
IoTpot [5]	Low	✓	V	x	✓
IoTcandyJar [6]	Medium	x	-	x	✓
SIPHON [7]	High	✓	R	x	x
ThingPot [8]	Medium	✓	V	✓	✓
HoneyCloud [9]	Low	✓	V	✓	✓
Costin et al. [10]	Low	✓	V	x	✓

traffic of its botnet. The advantages mentioned above conduce to hide malicious DHT flow with valid attack payload in the vast amount of normal DHT traffic, thus alleviating the detection and acquiring more time to quickly spawn botnets.

B. Investigation on IoT HoneyPots

IoT HoneyPot is distinguished from the traditional one by its focus on IoT malware. As Table II shows, we summarize the critical properties of six IoT HoneyPots. The front-end interaction capability and back-end implementation of them are closely related to the features of IoT malware.

1) *Front-end Interaction Capability*: The front-end interaction capability of IoT HoneyPot is mainly concerned with the generated response for a particular request that could trigger further attacks from IoT malware. IoT malware can utilize a certain protocol to identify and locate vulnerable devices. Only if IoT HoneyPot supports the specific protocol and generates the "right" response that malware is interested in can the attack and invasion be triggered. Therefore, IoT HoneyPots ought to serve plenty of protocols for capturing various unknown IoT malware and relevant traffic.

As far as the front-end interaction capability is concerned, existing IoT HoneyPots can be divided into three categories: high, medium and low interaction. Our evaluation is based on two criteria: the number of supported protocols and the quality of response packages.

IoTpot, for example, aims at the Telnet-based attack. It sets several pairs of weak login credentials for Telnet protocol only and can merely handle the Telnet-based reconnaissance from IoT malware. In contrast, IoTcandyJar is a high-interaction HoneyPot for its intelligent scanner and learner, which is able to collect the behaviors of all IoT devices on the Internet and utilize machine learning algorithms to select the best-learned response to attackers. Besides, regarded as a medium interaction HoneyPot, SIPHON deploys 85 geographically distributed device instances with a diverse set of IPs. It uses an SSH tunnel to forward the traffic from instances in cloud services to seven real IoT devices, including five physical cameras, one printer and one NVR, to realistically mimic several types (fewer than high-interaction HoneyPot) of interaction in a realistic way.

2) *Back-end Implementation*: The comprehensive IoT firmware and CPU architecture support contribute to the back-end capability of IoT HoneyPots. There are two options to implement the backend of IoT HoneyPot:

- Real system: different types of real IoT devices are deployed to provide a real execution environment.
- Virtual system: multiple types of vulnerable IoT firmware are emulated with different architectures using machine emulator and virtualizer (e.g., QEMU).

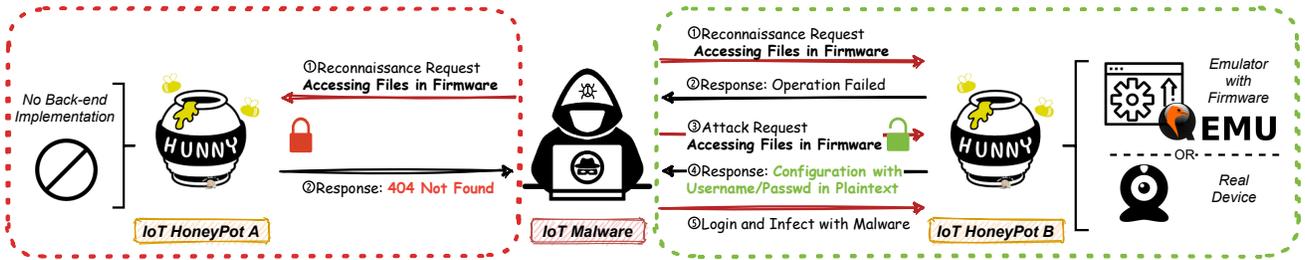


Fig. 1. CVE-2017-8225 - Pre-Authorize Information Leak (credentials) within The Custom HTTP Server of Wireless IP Camera (P2P) WIFICAM.

Both the real physical Honeypot and the virtual emulated Honeypot should have back-end implementation with various kinds of firmware and CPU architectures. To illustrate, Fig. 1 displays a typical exploit scenario of IoT malware, e.g., IoTReaper and Echobot. Without back-end implementation, IoT Honeypot_A fails to meet the condition of malware injection. On the contrary, IoT Honeypot_B is capable of attracting and deceiving attackers by virtue of firmware emulation or physical device deployment.

Specifically, IoT malware first conducts a reconnaissance request to identify the target or victim with “*http://target_ip/system.ini?loginuse=BAD_USRNAME&loginpas=BAD_PW*” (Step ① in Fig. 1). IoT Honeypot_A suffers from a lack of back-end implementation (i.e., missing *system.ini* existing in IoT firmware) and thus responds with HTTP status code 404 (Not Found). In other words, IoT Honeypot_A fails to cause malware infection and collect malware binary. Note that IoT Honeypot_B is equipped with a well-designed backend. Therefore, IoT Honeypot_B is able to handle the request by accessing the *system.ini* file and further respond to the attacker with a reasonable “Authorized Failed” (②). Later, since the access to the *system.ini* file are not correctly checked, IoT malware can bypass the authentication by providing an empty *loginuse* and an empty *loginpas* in the HTTP request URI (③). By doing so, IoT malware retrieves the clear-text configuration, including credentials, in the *system.ini* file (④) and then logs in with the confirmed username and password. Finally, it infects IoT Honeypot_B with the malicious binary of appropriate architecture and launches an attack (⑤).

III. INDUSTRIAL PRACTICE OF HONEYIoT

To deal with the heterogeneity of IoTs that IoT malware utilizes during the invasion, we stress the importance of the front-end interaction capability and the back-end implementation of IoT Honeypot. More precisely, providing the customized protocol servers and the back-end firmware emulation can create a more attractive Honeypot for IoT malware to intrude by either password cracking or exploit. Moreover, for the attack part of IoT malware, IoT Honeypot requires proper countermeasures to be reused. Inspired by these, we implement HoneyIoT, a primary and scalable IoT Honeypot framework, which aims at attracting IoT attacks and recording malicious behaviors with configurable vulnerabilities, firmware support and reset strategies.

A. HoneyIoT Design

Fig. 2 depicts the overview of HoneyIoT. At a high level, we run an operating system(e.g., OpenWrt) on an emulated

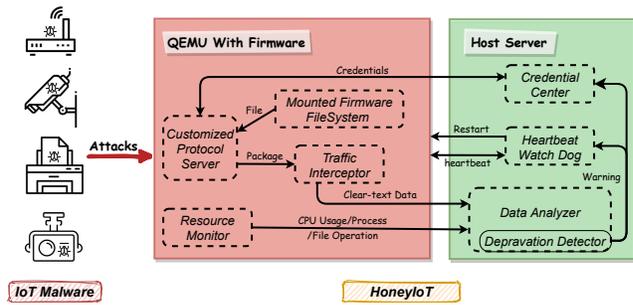


Fig. 2. Overview of HoneyIoT. Modules in the red rectangle take charge of the interaction with IoT malware holding configurable vulnerabilities and file system of firmware, while others in the green rectangle analyze and process the collected data related to invasion and attack of IoT malware.

CPU using QEMU. To handle IoT malware’s reconnaissance, HoneyIoT establishes the *Customized Protocol Server* module coupled with *Mounted Firmware File System* and *Credential Center* for better interaction capability. These extensible modules are conducive to mimic diverse IoT devices well by configurable specific response packets as vulnerable IoT devices do. Then *Traffic Interceptor* collects the request and response packets during the interaction while *Resource Monitor* records any change of CPU usage, process list and sensitive files. After that, *Data Analyzer* module defines malicious activities and generates warning reports based on empirical rules. In order to manage attacks from malware, HoneyIoT framework leverages *Heartbeat Watch Dog* module to reset the QEMU instance and internal modules. In Section II-A2, we categories attack methods of IoT malware into three different types: DDoS botnet, Mining and Phlashing. *Depravation Detector* informs *Heartbeat Watch Dog* and *Credential Center* of anomaly traffic flow or CPU usage to reset HoneyIoT and alternatively block the same login requests, thus alleviating the DDoS (i.e., avoid being recruited to the botnet) or Cryptocurrency Mining attack from already captured IoT malware. Apparently, it is unnecessary to repeat measurements of successful attacks from the same attacker. Besides, *Heartbeat Watch Dog* periodically send heartbeat packages to QEMU instance for early detection of Phlashing attack. In general, *Depravation Detector* together with *Heartbeat Watch Dog* reset QEMU in time whenever HoneyIoT turns out to be a captive or victim of IoT malware.

Scalability. As the heart of HoneyIoT framework, *Customized Protocol Server* enhances scalability and interaction capability by providing a number of configurable embedded device vulnerabilities that can be exploited by existing IoT malware. To be more specific, this module assembles prede-

finned protocol fields (e.g., Server: Linux, HTTP/1.1, DIR-300 Ver 2.12) into the response packet for a certain request (e.g., POST /command.php HTTP/1.1) to expose vulnerabilities. If needed, the *Mounted Firmware File System* offers back-end support of several IoT devices for generating valid bodies of response packets. Due to configurable vulnerabilities and back-end firmware support, HoneyIoT boasts the extendibility and interaction capability.

Reusability. *Depravation Detector*, *Heartbeat Watch Dog* and *Credential Center* make HoneyIoT more reusable. In our implementation, *Data Analyzer* employs heuristic strategies to define anomaly CPU usage and leverages existing detecting tools (e.g., Snort [11]) to detect the exploit behaviors in the traffic. Then, when the captive HoneyIoT is forced to carry out attacks(e.g., DDoS or Cryptocurrency-Mining), *Depravation Detector* inside *Data Analyzer* warns *Heartbeat Watch Dog* of the attack, which can timely reset the QEMU instance to the previous benign state. The same reset operation may occur when three failed heartbeat checks in a row. Besides, *Credential Center* maintains valid login credentials. It allows all login attempts by default and alternatively modifies login policy when HoneyIoT is cracked to block repeated malware injection. Therefore, HoneyIoT improves the reusability.

B. Preliminary Result

We deployed HoneyPot on three geographically distributed virtual machines from Vultr and Alibaba Cloud. We also customized HoneyIoT with three protocol support, including SSH, Telnet and HTTP, as well as two kinds of router firmware, i.e., D-Link 850L and Netgear WNAP320. Besides, we provide a number of device vulnerabilities [12] exploited by existing IoT malware, such as IoTReaper. During a 7-day real-world industrial experiment, HoneyIoT observed over 12,500 malicious connections and 3,423 distinct login attempts with weak passwords. Here, we share some interesting findings from the captured traffic in the following section:

- 1) **Password cracking keeps pace with the times.** We found that 57 types of weak passwords contain the “2020” string, such as asdf2020, 1q2w3e2020, Password#2020, admin2020 and HuaWeiN2020, etc. It can be seen that the existing active malware employs fairly new cracking dictionaries. It is more likely that these kinds of malware are recently developed or redesigned as new variants of previous IoT malware.
- 2) **Stronger safeguards are needed for Virtual Server Providers.** We analyze the source IPs of requests that have connected to HoneyIoT more than 190 times in 7 days. According to AbuseIPDB¹, the confidence of abuse of these malicious IP addresses can reach up to 70 percent on Jan. 2, 2021. What is noteworthy is that some IPs of attackers is affiliated with Virtual Dedicated Server(VDS) Providers like PIN², a Russian company (see Table III), observed by reverse DNS resolution (rDNS). It is therefore reasonable to infer that some virtual machines of cloud service providers are abused as network agents for malware.

¹<https://www.abuseipdb.com/>

²<https://pinspb.ru/>

TABLE III
MALICIOUS IPs OF A RUSSIAN VDS PROVIDER(PIN)

Malicious IP	Connections	Confidence of Abuse	Associated Domain Name
5.188.86.206	190	72%	pinspb.ru
5.188.87.49	196	71%	pinspb.ru
5.188.86.169	198	72%	pinspb.ru
5.188.87.60	223	72%	pinspb.ru
5.188.86.216	232	70%	pinspb.ru
5.188.86.221	247	72%	pinspb.ru
5.188.86.212	254	72%	pinspb.ru

IV. CONCLUSION

In this paper, we make a comprehensive analysis of IoT malware to identify the dilemma of IoT HoneyPots and further refine it. By analyzing the invasion and attack methods of real-world IoT malware, we realize that the front-end interaction capability and the back-end implementation are of great significance for IoT HoneyPots. Therefore, we propose HoneyIoT, a primary and scalable IoT HoneyPot framework. The experiment results manifest HoneyIoT is competent to collect the traffic of real-world IoT attacks. In the future, we would make the best use of the scalability and reusability of HoneyIoT by customizing and deploying it according to different malware families, which can eventually achieve more effective deployments of IoT HoneyPots.

ACKNOWLEDGMENT

This research is sponsored in part by the NSFC Program (No. 62022046, U1911401, 61802223), National Key Research and Development Project (Grant No. 2019YFB1706200), the Huawei-Tsinghua Trustworthy Research Project (No. 20192000794).

REFERENCES

- [1] M. Antonakakis, T. April *et al.*, “Understanding the mirai botnet,” in *USENIX Security*. USENIX Association, 2017, p. 1093–1110.
- [2] C.-W. Tien, S.-W. Chen *et al.*, “Machine learning framework to analyze iot malware using elf and opcode features,” *Digital Threats: Research and Practice*, vol. 1, no. 1, Mar. 2020.
- [3] F. Ding, H. Li *et al.*, “Deeppower: Non-intrusive and deep learning-based detection of iot malware using power side channels,” in *ASIA CCS*. Association for Computing Machinery, 2020, p. 33–46.
- [4] A. Turing and H. Wang, “Mozi, another botnet using dht,” <https://blog.netlab.360.com/mozi-another-botnet-using-dht/>, 2019.
- [5] Y. M. P. Pa, S. Suzuki *et al.*, “Iotpot: Analysing the rise of iot compromises,” in *WOOT*. USA: USENIX Association, 2015, p. 9.
- [6] T. Luo, Z. Xu *et al.*, “Iotcandyjar: Towards an intelligent-interaction honeypot for iot devices,” *Black Hat*, 2017.
- [7] J. D. Guarnizo, A. Tambe *et al.*, “Siphon: Towards scalable high-interaction physical honeypots,” in *CPSS*. Association for Computing Machinery, 2017, p. 57–68.
- [8] M. Wang, J. Santillan *et al.*, “Thingpot: an interactive internet-of-things honeypot,” *CoRR*, vol. abs/1807.04114, 2018.
- [9] F. Dang, Z. Li *et al.*, “Understanding fileless attacks on linux-based iot devices with honeycloud,” in *Mobisys*. Association for Computing Machinery, 2019, p. 482–493.
- [10] A. Costin and J. Zaddach, “Iot malware: Comprehensive survey, analysis framework and case studies,” *BlackHat USA*, 2018.
- [11] M. Roesch, “Snort: Lightweight intrusion detection for networks,” in *LISA*, D. W. Parter, Ed. USENIX, 1999, pp. 229–238.
- [12] SSD Secure Disclosure, “D-link 850l multiple vulnerabilities (hack2win contest),” <https://ssd-disclosure.com/ssd-advisory-d-link-850l-multiple-vulnerabilities-hack2win-contest/>, 2017.