Poster: IoTcube: An Automated Analysis Platform for Finding Security Vulnerabilities

Seulbae Kim, Seunghoon Woo, Heejo Lee, Hakjoo Oh
Department of Computer Science and Engineering
Korea University
Seoul, Korea
{seulbae, seunghoonwoo, heejo, hakjoo_oh}@korea.ac.kr

***Abstract*—Although the quantity of services and devices regarding the Internet of Things (IoT) is consistently increasing, not many people are aware that software vulnerabilities are also proliferating at an alarming rate along with the spread of IoT. In addition, for people without security backgrounds, defending their devices against these vulnerabilities is also a huge challenge. IoTcube, an automated analysis platform for finding security vulnerabilities in the IoT devices, is developed to be a guidance system for any people with or without security expertise.**

I. Introduction

As the use of IoT devices is exponentially increasing, vulnerabilities residing in IoT software are also proliferating accordingly. In July 2014, a Hewlett-Packard study [1] revealed that 70 percent of IoT devices are vulnerable to attacks. In 2017, a global study [2] conducted by Hewlett Packard Enterprise Aruba found that 84 percent of organizations have experienced at least one IoT-related security breach. A report by Juniper Research pointed out that the number of IoT devices will increase by 200 percent to 46 billion units by 2021. Increased number of connected IoT devices will leave more and more users wide open to growing number of unidentified threats.

Unfortunately, not every IoT manufacturer has its own security specialist who can examine and fix security weaknesses. To aggravate the situation, many IoT systems (i.e., the operating system and utilities running on top of it) generally include the Linux kernel and a number of open source software which are too large to be manually inspected by a security specialist. Although a few vulnerability detection techniques have been proposed [3], [4], these techniques often require too much expertise, and some of them have performance issues (e.g., high false-positive rate or low scalability).

With the expanding volume of IoT devices and associated vulnerabilities, an automated approach for scalable and reliable vulnerability analysis is required. Center for Software Security and Assurance[1] (CSSA) was founded in 2015 to accommodate the desperate needs for automated vulnerability detection and verification that even non-security specialists can use with ease, and to build a global community for securing IoT system software. CSSA developed *IoTcube* which comprises of four distinct teams specializing in black-box testing, white-box testing, network testing and platform establishment, to provide a comprehensive vulnerability testing environment (Fig. 1).

[1]Led by professor Heejo Lee, located in Korea University Seoul Campus, available at http://cssa.korea.ac.kr

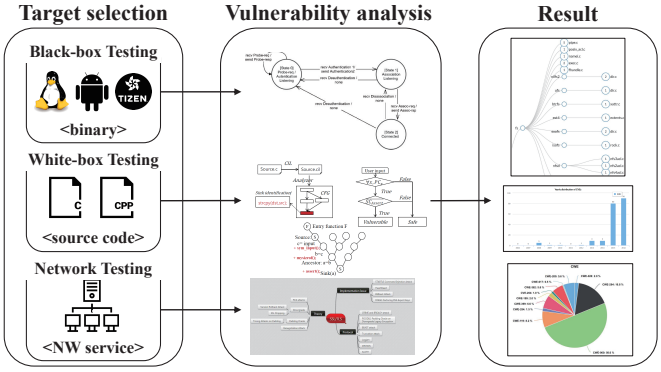

Fig. 1: Overview of automated vulnerability testings.

II. IoTcube

With the mission of "providing everyone with various easy-to-use analysis to discover vulnerabilities in software and hardware systems", CSSA developed an automated analysis platform called *IoTcube*[2], and released it to the public on April 19, 2016. For the purpose of providing highly influential automated vulnerability testing platform to the users of varying levels of security expertise (e.g., from security experts to normal users), *IoTcube* is built upon three following strategies:

1) Focusing on four major study fields: To guarantee an effective detection and analysis of various kinds of vulnerability from various devices under different conditions, CSSA runs four specialized research teams: Black-box testing; White-box testing; Network testing; and Platform team. Black-box testing team mainly focuses on detecting vulnerabilities from program binaries and protocol implementations. White-box testing team analyzes and verifies program source code to find vulnerabilities. This approach can be applied in the software production state, to check the existence of known and unknown vulnerabilities before compiling and releasing products. Network testing team studies approaches to automate the analysis of network protocol vulnerabilities. It enables users to test designated servers against critical vulnerabilities, e.g., HeartBleed vulnerability. Platform team devotedly studies better ways to show testing results to users in order to provide the best user experience. They design user interface and establish an interaction model by applying Human-Computer Interaction interface design theory.

[2]https://iotcube.net

2) Conducting international joint research: Globally leading research groups from Carnegie Mellon University, Oxford University, and ETH Zurich are cooperating to develop and integrate leading technologies into *IoTcube*.
3) Community Connection and Consultation: With the vulnerabilities and insights gathered through *IoTcube*, CSSA conducts seminars and have regular technical exchanges with one of the best hacking teams, PPP (Plaid Parliament of Pwning) of Carnegie Mellon University, and Korean hacking team CodeRed. In addition, CSSA also seeks for the involvement of Inc0gnito which is an association of security clubs in 12 major universities in South Korea.

III. Automated Vulnerability Testings

A. Black-box Testing

Black-box testing team develops vulnerability analysis tools based on dynamic black-box testing.

1) Bluetooth fuzzer (bfuzz): Bluetooth Fuzzer (bfuzz) is a tool for discovering implementation error of Bluetooth-enabled devices by using smart and stateful fuzzing technique. The fuzzing engine automatically generates possible vulnerable inputs regarding four kinds of Bluetooth protocol specification, i.e., L2CAP, OBEX, RFCOMM, and SDP, and it is able to discover unknown vulnerabilities resulted from implementation errors residing in Bluetooth stack of target devices.

2) Wi-Fi Fuzzer (wfuzz): Wi-Fi Fuzzer (wfuzz) is a tool for discovering implementation error of 802.11-enabled devices by using smart and stateful fuzzing technique. It automatically generates possible vulnerable inputs regarding the 802.11 protocol specification. wfuzz is able to discover unknown vulnerabilities resulted from implementation errors residing in 802.11 stack of target devices.

3) Command Line Option Inference (cloif): The Command Line Option Inference (cloif) tool automatically discovers possible input options which can be taken by a binary. By utilizing dynamic instrumentation engine (DynamoRIO) developed by Google and MIT, it effectively finds as many options accepted by a target binary as possible.

B. White-box Testing

White-box testing team focuses on developing static-analysis oriented vulnerability detection and verification tools.

1) Vulnerable code clone detection (hmark): Vulnerable Code Clone Detection (VCC Detection) [5] is an approach for the scalable detection of vulnerable code clones, which is capable of detecting security vulnerabilities in large software programs efficiently and accurately. This approach bases on the intuition that vulnerable code is reused through inadvertent code cloning. It currently has signatures of more than 5.6 K vulnerable functions that address 1,764 unique CVEs (Common Vulnerability Enumerations) in the database of *IoTcube*.

2) Potential vulnerability verification (ctest): Potential Vulnerable Code Verification (PVC Verification) [6] is an approach for C source code vulnerability testing. By using ctest, users can verify vulnerabilities of their C program by applying given test strategies.

C. Network Testing

Network testing team aims to detect and conduct analysis of network code vulnerability and network protocol vulnerability-related issues.

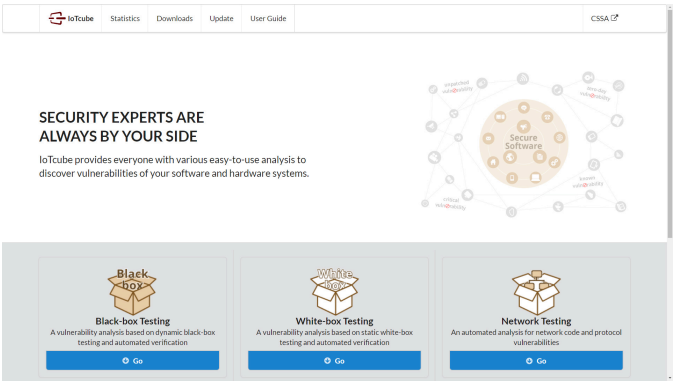

Fig. 2: The main page of *IoTcube*.

1) Network security scanner (nscan): Network Security sCanner with Automatic detectioN (nscan) [7] is a tool for scanning TLS vulnerabilities in target web server. It can detect seven critical TLS vulnerabilities such as HeartBleed, POODLE, and DROWN, and four weak cryptographic primitives including MD5 and SHA1 cryptographic algorithm.

IV. Result and Statistics

IoTcube was released to the public on April 19, 2016 via `https://iotcube.net` (Fig. 2). 2.7 K different users conducted about 5 K testings on the platform for a year, and discovered more than 164 K unique vulnerabilities.

V. Conclusion

IoTcube is an effective solution for ever-growing IoT software vulnerabilities. Users are able to examine a wide range of targets, e.g, program binary, source code, or even running web servers, through a few clicks through IoTcube, and get a reliable analysis result.

Acknowledgment

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.R0190-16-2011, Development of Vulnerability Discovery Technologies for IoT Software Security).

References

[1] K. Rawlinson, "HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack," http://www8.hp.com/us/en/hp-news/press-release.html?id=1744676#.WRhhMGjygx8, July 2014, Accessed: 2017-05-01.
[2] Hewlett Packard Enterprise, "The Internet of Things: Today and Tomorrow," http://www.arubanetworks.com/assets/eo/HPE_Aruba_IoT_Research_Report.pdf, March 2017, Accessed: 2017-05-01.
[3] M. Pistoia, S. Chandra, S. J. Fink, and E. Yahav, "A survey of static analysis methods for identifying security vulnerabilities in software systems," *IBM Systems Journal*, vol. 46, no. 2, pp. 265–288, 2007.
[4] B. Liu, L. Shi, Z. Cai, and M. Li, "Software vulnerability discovery techniques: A survey," in *Multimedia Information Networking and Security (MINES), 2012 Fourth International Conference on*. IEEE, 2012, pp. 152–156.
[5] S. Kim, S. Woo, H. Lee, and H. Oh, "VUDDY: A Scalable Approach for Vulnerable Code Clone Discovery," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017.
[6] H. Li, J. Oh, H. Oh, and H. Lee, "Automated source code instrumentation for verifying potential vulnerabilities," in *IFIP International Information Security and Privacy Conference*. Springer, 2016, pp. 211–226.
[7] J. Jeong, H. Kwon, H. Shin, and J. Hur, "A practical analysis of tls vulnerabilities in korea web environment," in *International Workshop on Information Security Applications*. Springer, 2016, pp. 112–123.