Demo: An Emulator-based Active Protection System against IoT Malware

Shin-Ming Cheng and Sheng-Hao Ma
Dept. CSIE, National Taiwan University of Science and Technology, Taipei, Taiwan
smcheng@mail.ntust.edu.tw and aaaddress1@chroot.org

***Abstract*—This demonstration presents an emulator-based active protection system particularly for IoT malware identification and blocking. The key component of our system is a new design of an application loader and an emulating engine based on Unicorn. We demonstrate using IoT network consisting of IoT gateway and IoT devices where the proposed system can be enabled in face of the infamous Mirai attack. We show that with the aid of emulation engine, malicious commands triggered by Telnet and SSH-based IoT malware can be identified and blocked effectively and efficiently while eliminating the possibility of virtual machine escalation.**

***Index Terms*—active protection, application loader, emulating engine, IoT malware, Unicorn**

I. Introduction

The large-scale infection of IoT malware caused severe damage and received lots of attentions. To prevent poorly configured IoT devices from being compromised via weak login credentials or vulnerabilities, and being exploited as bots to launch DDoS attack, IoT honeypots are designed to capture IoT malwares and to analyze their signatures. It is believed that high-interactive IoT honeypots could engage malicious user for longer sessions and capture malwares more effectively. However, it might not be worth to implement all system functionalities for Telnet or SSH-based IoT malwares. Moreover, the analysis of downloaded malware introduces extra delay for the control of malware propagation. As a result, a more effective and efficient protection approach for simple IoT malware is necessary.

II. System Description

As shown in Fig. 1, our framework consists of two main services: *monitoring* and *detection services*. The *monitoring service* identifies if the Telnet and SSH-related process contain wget, curl, or echo command, which might result in additional file downloading. If it does, the downloaded file will be sent into the application loader in *detection service*. The service could read an executable file as an OS process, emulates the whole interactions, and identify the malicious behavior of a process.

The application loader is developed on the top of emulating engine, which is implemented following USERCORN. As a result, the largest Linux Syscall Interrupts are applied and MIPS, x86, and ARM architectures are supported. To prevent the executing file from causing virtual machine escalation to infect the real world, our emulator ensures the isolation of CPU, memory management, and OS. With the aid of

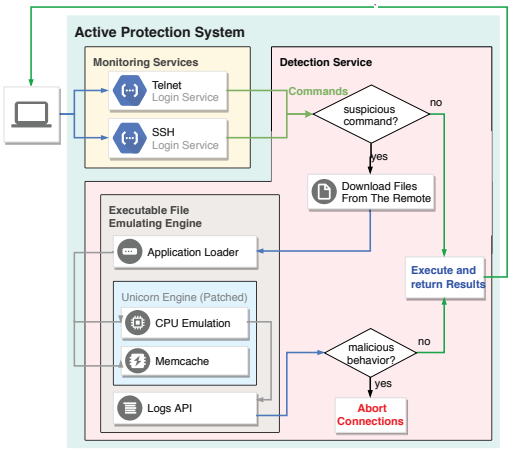

Fig. 1. Functional Diagram of Active Protection System.

Unicorn engine based on QEMU, we implement a new virtual CPU, which could interpret machine code into assembly codes by the x86, ARM, or MIPS instruction set and apply the computing ability of each assembly code. Moreover, the APIs from Unicorn engine are applied for the development of the isolated memory management. Regarding isolated OS, we just implement the functionality of Linux syscall so that the interactions between process can be appropriated emulated. By establishing the system on the IoT intermediate nodes such as router or gateway, inappropriate commands performed by the IoT malwares can be blocked, thereby protecting the IoT devices in the local network actively.

III. Demonstration

In our demonstration, we set up an simple IoT home environment consisting of an IoT intermediate node (i.e., a router) and several IoT devices with weak login credentials. A malicious user tried to apply the infamous Mirai sample to attack the IoT networks and our active protection system is installed on the intermediate node.

With the aid of the proposed system, every commands launched from Telnet connection will be checked if any downloading behavior or executable file inside. If it happens, our solution will launch the malware engine to justify it is malicious, and avoid the downloading action or keep the commands launched. As a result, the vulnerable router still stable against the IoT malware.

DOI 10.1109/SPW.2019.00038