| | |
|---|---|
| Title | Automatic Stub Generation for Dynamic Symbolic Execution of ARM binary |
| Author(s) | Nguyen, Thi Van Anh |
| Citation | |
| Issue Date | 2021-09 |
| Type | Thesis or Dissertation |
| Text version | author |
| URL | http://hdl.handle.net/10119/17543 |
| Rights | |
| Description | Supervisor:小川　瑞史, 先端科学技術研究科, 修士（情報科学） |

# Automatic stub generation for dynamic symbolic execution of ARM binary

1910407 Nguyen Thi Van Anh

In a recent survey, it is reported that the number of IoT attacks in 2020 has been rose to 66% compared to the previous year. Due to the vast number of IoT devices with weak authentication and lack of security protection ability, IoT devices have become easy targets for exploitation. With the rapidly increasing number of IoT devices, even when the computing power of each IoT device is low, they can collaborate for making large-scale attacks (e.g., DDoS, and crypto-jacking). Therefore dealing with IoT malware has become more and more urgent and necessary. There are many existing approaches for analyzing malware including static analysis and dynamic analysis. However, disassemblers can be easily cheated by obfuscation techniques, and dynamic analyses can be detected and prevented by VM awareness, anti-debugging, and trigger-based behavior. To bypass obfuscation techniques, especially for indirect jump resolution, it is necessary to apply Dynamic Symbolic Execution to reconstruct malware execution trace.

ARM is a processor family, which is most popularly used for IoT devices. In previous work, a DSE tool for ARM Cortex-M - CORANA was preliminarily built by extracting ARM formal semantics from natural language descriptions. However, malware frequently runs in both user mode and kernel mode, and they also connect with other external systems (e.g, C&C servers, or other end-user devices). The result of the external calls might affect deeply the analysis, especially in the presence of anti-debugging or trigger-based code. An approach is to prepare API Stub of system calls to interact with the external environment. Automatically generation of API Stub can help the symbolic execution engine produce meaningful execution traces while reducing the cost of manual API implementation.

We target ARM on Linux, where its API specification is available in the Linux Manual Page. This thesis proposed an approach to systematically generate Linux API Stub from the C library function interface description. For each library function, first, we apply pattern matching to retrieve the information on its name, parameters, and return type. After that, we using predefined type conversion rules to statically decided on the types of parameters. Then, three kinds of Java classes of the target function which are structure definition class, interface-mapped class, and API Stub class are generated. By 1659 collected API descriptions, we are able to produce 1129 API Stubs and 267 structure definition classes. We also proposed using serialization to handle the execution of multiple processes. To demonstrate the ability of CORANA after adding the generated API Stubs to support external calls (CORANA/API), we performed a detailed analysis on a Mirai sample using the tool. The result shows that CORANA/API is able to trace real-world IoT malware samples and is resilient against several obfuscation techniques, which overcomes the existing DSE tools, e.g., angr.

***Keywords***— IoT malware, malware analysis, Dynamic Symbolic Execution, ARM Cortex-M