Poster: Visualization Result of String-based CPU Architecture Independent IoT Malware Clustering

Yutaro Osako*, Toshihiro Yamauchi†, Katsunari Yoshioka‡, Takuya Fujihashi*, Takashi Watanabe*, Shunsuke Saruwatari*
*Graduate School of Information Science Technology, Osaka University, Japan
†Graduate School of Natural Science and Technology, Okayama University, Japan
‡Graduate School of Environment and Information Sciences, Yokohama National University, Japan

***Abstract*—This paper presents a visualization result of unsupervised clustering using IoT malware datasets acquired by IoTPOT [4], [5]. The problems of IoT malware are a wide variety of CPU platforms and the rapid evolve of variants from the same family. For supporting the rapid analysis of IoT malware, this paper proposes a CPU-independent string-based clustering method. In the analysis process, we found 25 samples that contained a unique string among 4000 samples, and we succeeded in finding the malware's source code from a web search using the unique string.**

I. Introduction

Network devices have been installed everywhere, and it has become common for things to be connected to the network, making our lives more convenient. As network devices and network-connected objects become targets for attackers, they are becoming hotbeds of damage on an unprecedented scale, and it is essential to take action against malware. In this paper, we refer to malware that infects network devices and IoT devices other than PCs and smartphones as IoT malware. In 2016, IoT malware Mirai [1], [3] caused a DDoS attack of 665 Gbps, which was the largest traffic at that time.

However, IoT malware differs from existing PC malware in the following two ways, (1) various CPU architecture, (2) many variants from the same family. The characteristics make it difficult to determine the nature of the IoT malware. IoT devices have a variety of CPU architectures. The attackers generate multiple binaries with different characteristics from a single source code for the different CPU architectures. Adding "a large number of variants from the same family" to "the various CPU architectures" makes it even more difficult to identify IoT malware. Vulnerabilities in IoT devices are often rudimentary, such as default passwords. The rudiment vulnerability makes the IoT malware behave simply and makes it easy to generate variants.

II. String-based Malware Clustering Algorithm

From the discussion in Section I, this paper proposes a string-based malware clustering algorithm (SMCA) to classify variants of IoT malware efficiently without supervised learning. SMCA classifies IoT malware without depending on the CPU architecture using only the strings contained in the IoT malware binary. In addition, we can use the SMCA if there is no correct label because SMCA is unsupervised learning. We can also generate a new YARA rule from strings representing a particular cluster feature obtained from SMCA.

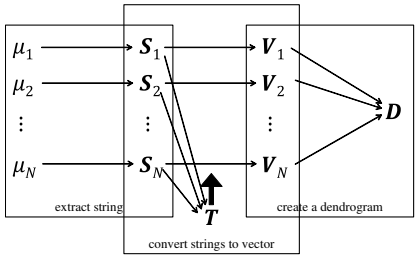


Fig. 1. String-based malware clustering algorithm

Figure 1 shows the overview of the proposed method. The SMCA consists of extracting strings from each binary, converting strings to vectors, and performing hierarchical clustering. N is the number of binary files of the malware to be clustered, μ_i is the binary file of each malware sample, $\boldsymbol{S}_i$ is the table of strings extracted from each malware sample μ_i by the string extraction, $\boldsymbol{T}$ is the set of strings extracted from all malware, $\boldsymbol{S}_i$ is a table of strings extracted from each malware μ_i by string extraction, $\boldsymbol{T}$ is a set of strings extracted from all malware μ_i and duplicates removed, $\boldsymbol{V}_i$ is a vector associated with each malware sample μ_i, and $\boldsymbol{R}$ is the tree diagram data generated from $\boldsymbol{V}_1, \boldsymbol{V}_2, \cdots, \boldsymbol{V}_N$ by the hierarchical clustering algorithm.

III. Results

A. Datasets

We evaluated the proposed method, SMCA, using an IoTPOT malware binary dataset A [5] of 4000 IoT malware samples collected from October 2, 2016, to October 2, 2017, by IoTPOT [4] in Yoshioka Laboratory, Yokohama National University, and nine samples compiled from the Mirai source code available on [2], to obtain a total of 4009 samples. We used the file command to examine the CPU architectures of the 4009 malware samples and found 937 MIPS, 914 ARM, 577 Intel 80386, 354 PowerPC, 320 x86-64, 295 Motorola m68k, 300 SPARC, 311 Renesas SH, and one Text. Although most binaries are ARM and MIPS, which are commonly used in embedded Linux, there are more than 200 Renesas SH binaries. Each of the 4009 binaries was distinguished by a hash value obtained by applying MD5 to the entire binary, and it was confirmed that the binaries were different. In addition, 1407 of the 4009 binaries had their symbol information deleted by the stripping.

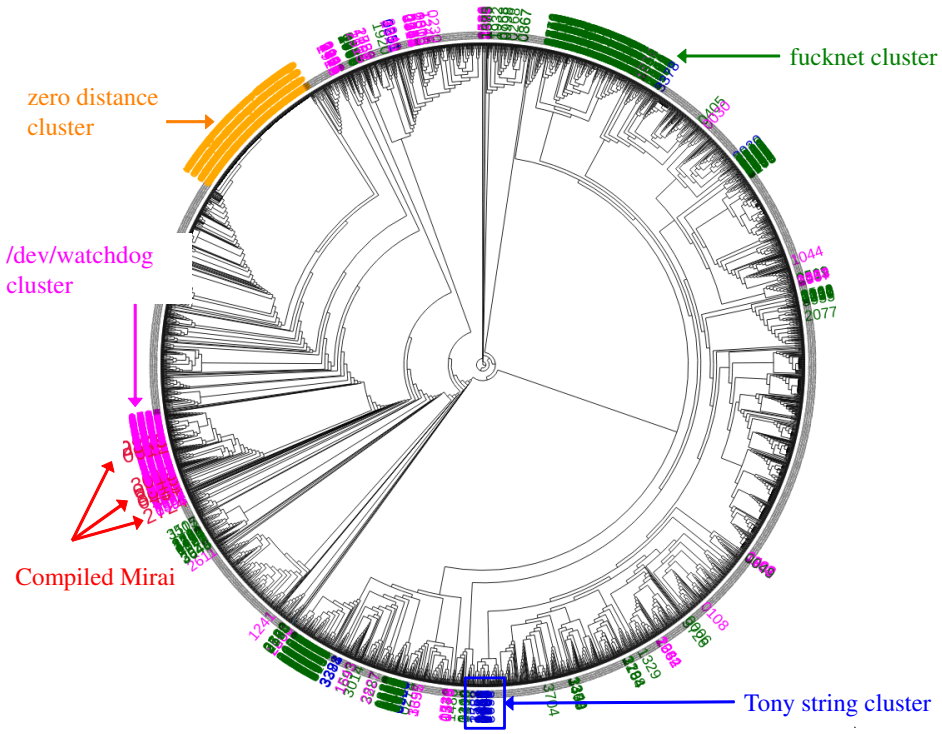


Fig. 2. Visualization result of clustering

B. Visualization

Figure 2 shows a visualization result of SMCA using the dataset in Section III-A. The center is the root of the classification tree, and the outermost circle is the leaf. The IDs (0–4008) assigned to each binary are displayed on the leaves. In Figure 2, we focus on compiled Mirai, a possible Mirai that contains the string `/dev/watchdog`, and a possible BASHLITE that contains the string `fucknet` in our analysis. The IDs themselves are not visible due to a large number of leaves, so the IDs, which are our focus, are colored. In order to investigate how different CPU architectures with the same source code are classified by the proposed method, we checked the location of Mirai on the classification tree compiled in nine different CPU environments: ARM static link, ARM dynamic link, Motorola m68k, MIPS MSB, MIPS LSB, PowerPC, Renesas SH, SPARC, and x86. The red-lettered IDs in Figure 2 indicate the compiled Mirai. We can see that the proposed SMCA succeeded to classify binaries of different CPU architectures into close positions if the source codes are the same.

In evaluating the proposed method, we found 25 samples that contained a specific string. In this paper, we refer to this string as the Tony string. The blue IDs in Figure 2 are the samples that contain the Tony string. When surveying the Tony string, some search engines gave us zero hits. However, one search engine found only one page related to Pastebin. We found the source code of malware that contained the Tony string from the found page.

IV. Conclusion

This paper proposed a hierarchical clustering algorithm using strings and showed the visualization results using the algorithm with an IoTPOT malware binary dataset A [5]. We found the Tony strings that seemed to be included in specific malware. Currently, we are working on the handling of stripped binaries, a detailed comparison with VirusTotal output results, and auto-generation of YARA rules.

Acknowledgment

This study was supported by JST, PRESTO Grant Number JPMJPR2032, JPMJPR1938, and NTT Corporation. A part of this research was conducted in ”MITIGATE” project among ”Research and Development for Expansion of Radio Wave Resources (JPJ000254)”, supported by the Ministry of Internal Affairs and Communications, Japan.

References

[1] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, “Understanding the mirai botnet,” in *Proceedings of the USENIX Security Symposium (USENIX Security’17)*, Vancouver, BC, 2017, pp. 1093–1110.

[2] J. Gamblin, “Mirai-source-code,” https://github.com/jgamblin/Mirai-Source-Code.

[3] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, “DDoS in the ioT: Mirai and other botnets,” *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[4] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, “IoTPOT: Analysing the rise of IoT compromises,” in *Proceedings of the USENIX Workshop on Offensive Technologies (USENIX WOOT’15)*, Aug. 2015.

[5] Yokohama National University, “IoT honeypot: Malware binaries dataset A,” https://sec.ynu.codes/iot/available_datasets#1.

A Visualization Result of String based CPU Architecture Independent IoT Malware Clustering

Yutaro Osako[1], Toshihiro Yamauchi[2], Katsunari Yoshioka[3],
Takuya Fujihashi[1], Takashi Watanabe[1], Shunsuke Saruwatari[1]

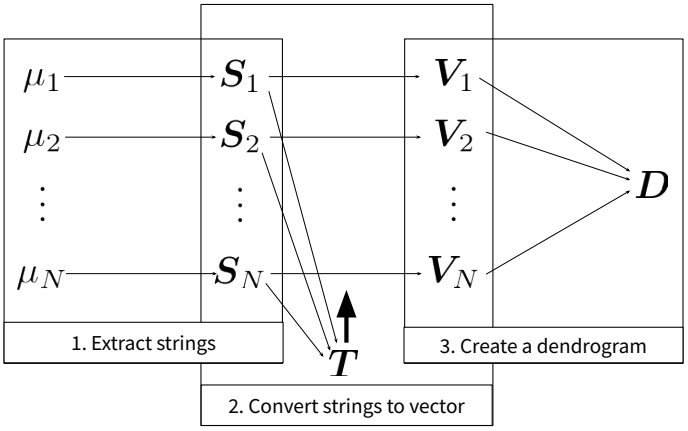
1. Graduate School of Information Science Technology, Osaka University
2. Graduate School of Natural Science and Technology, Okayama University
3. Graduate School of Environment and Information Sciences, Yokohama National University

Background

- Network devices and network-connected objects are becoming hotbeds of cyber attacks.
- The problems of IoT malware are a wide variety of CPU platforms and the rapid evolve of variants from the same family.
- **For supporting the rapid analysis of IoT malware, we propose a CPU-independent string-based clustering method.**

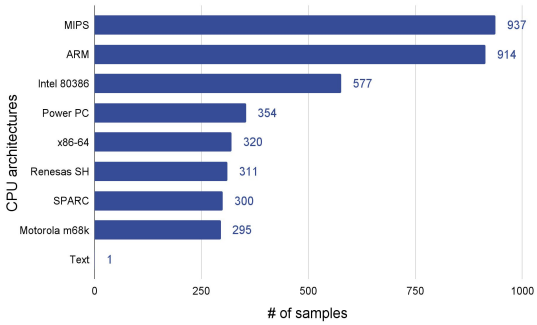
Clustering method

String-based Malware Clustering Algorithm (SMCA)



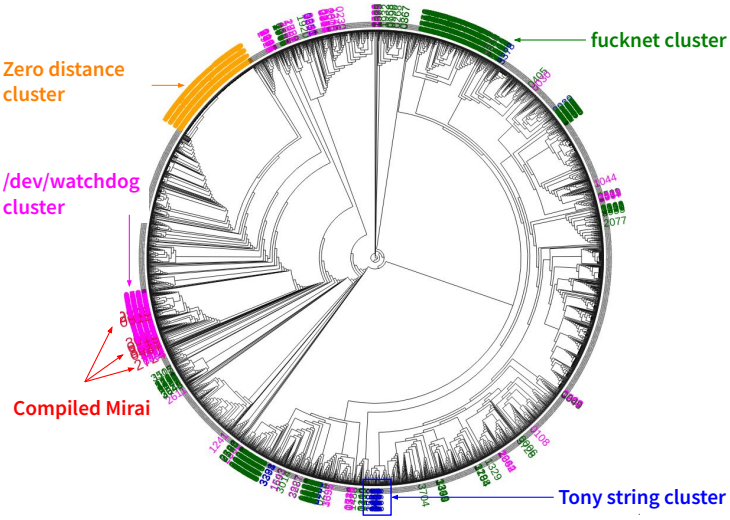
Dataset

- IoTPoT malware binary dataset A [1]
  - 4,000 samples
- Compiled Mirai [2]
  - Additional nine samples as ground truth data



[1] Yokohama National University, "IoT honeypot: Malware binaries dataset A," https://sec.ynu.codes/iot/available datasets#1
[2] J. Gamblin, "Mirai-source-code," https://github.com/jgamblin/Mirai-Source-Code

Visualization Result



- We checked the location of the compiled Mirai to investigate how different CPU architectures with the same source code are classified by the SMCA.
- **/dev/watchdog cluster**
  - Mirai contains a string */dev/watchdog*.
  - Compiled Mirai binaries are also classified into this cluster.
- **fucknet cluster**
  - BASHLITE(Gafgyt) contains a string *fucknet*.
- **Tony string cluster**
  - A unique string beginning with Tony.
  - The binaries containing this string make a cluster.
  - We found the malware's source code from a web search using the unique string.
- **Zero distance cluster**
  - The cluster possibly consists of stripped or obfuscated binaries.

Conclusion / Future Work

- **We found a malware's source code by carrying out a web search using an unique string from the clustering result.**
- Currently, we are working on the handling of stripped binaries, a detailed comparison with VirusTotal output results, and auto-generation of YARA rules.