International Journal of Internet, Broadcasting and Communication Vol.9 No.1 24-28 (2017)
https://doi.org/10.7236/IJIBC.2017.9.1.24

IJIBC 17-1-4

Probabilistic Analysis of Code-Reuse Attacks and Defenses in IoT

Jun-Won Ho

Department of Information Security
Seoul Women's University
621 Hwarangro, Nowon-Gu, Seoul, South Korea
jwho@swu.ac.kr

Abstract

In the Internet of Things (IoT), resource-limited smart devices communicate with each other while performing sensing and computation tasks. Thus, these devices can be exposed to various attacks being launched and spread through network. For instance, attacker can reuse the codes of IoT devices for malicious activity executions. In the sense that attacker can craft malicious codes by skillfully reusing codes stored in IoT devices, code-reuse attacks are generally considered to be dangerous. Although a variety of schemes have been proposed to defend against code-reuse attacks, code randomization is regarded as a representative defense technique against code-reuse attacks. Indeed, many research have been done on code randomization technique, however, there are little work on analysis of the interactions between code randomization defenses and code-reuse attacks although it is imperative problem to be explored. To provide the better understanding of these interactions in IoT, we analyze how code randomization defends against code-reuse attacks in IoT and perform simulation on it. Both analysis and simulation results show that the more frequently code randomizations occur, the less frequently code-reuse attacks succeed.

Keywords: *Probabilistic analysis; Code-reuse attack; IoT; Security*

1. Introduction

In IoT, smart devices are usually connected through wireless interface. Thus, they are vulnerable to wide range of attacks that could be generated and propagated through wireless networks. In particular, code-reuse attacks [5] are very deleterious to smart devices in the sense that attacker could perform malicious activities inside target devices by tactfully reusing the normal codes stored in them. Since the normal codes in target devices are reused, it would be likely difficult to detect the reused normal codes. To fight against code-reuse attacks, a variety of defense schemes have been developed in [1], [2], [3], [4].

Although these schemes mainly focus on how to defend code-reuse attacks, they do not separately deal with the probabilistic analysis of the interactions between code-reuse attacks and defenses. To mitigate this

Manuscript Received: Dec. 22, 2016 / Revised: Dec. 28, 2016 / Accepted: Jan. 20, 2017
Corresponding Author: jwho@swu.ac.kr
Tel: +82-2-970-5607, Fax: +82-2-970-5981
Department of Information Security, Seoul Women's University

problem, we propose a probabilistic analysis of the interactions between code randomization defenses and code-reuse attacks in IoT. The key idea of code randomization [2, 3] is to randomize the addresses of code segments in IoT devices and make it difficult for the attacker to reuse the addresses of codes. The reason why we choose code randomization as a defense technique is because it is regarded as a typical defense scheme. We first apply poisson process for the analysis of the interactions between code randomization defender and code-reuse attacker and then perform simulation on that analysis. From simulation results, we find that the fraction of successful code-reuse attacks is measured as 91.586% and 0.003% on an average when code-reuse attack time is 10 simulation seconds and the number of times of code randomization occurrence is 10.848 and 1004.94 on an average in 1000 simulation seconds, respectively.

2. Analysis and Evaluation of the Interactions between Code Randomizations and Code-Reuse Attacks

Attacker would not like to mount code-reuse attacks against IoT devices only once, but repeatedly launch these attacks. To defend the system against the persistent code-reuse attacks, defender also should recurrently perform code randomization defenses in IoT devices. Hence, analysis on the interactions between attacker and defender is required to be examined. A poisson process is suitable for fulfilling this analysis because it is appropriate to model when code randomization occurs in IoT devices.

We model the occurrence of code randomization as poisson process with parameter α. Accordingly, inter-occurrence time between two consecutive code randomizations follows i.i.d. exponential distribution with parameter α. Let us define R_1 as the time of the first occurrence of code randomization. We define R_k as an inter-occurrence time between the (k−1)th code randomization and the kth code randomization (k≥2). Assume that attacker is able to predict when code randomization occurs and thus he is capable of figuring out every inter-occurrence time of code randomizations. Moreover, we assume that it takes τ time for attacker to succeed in a code-reuse attack. If attacker wants to take benefits from code-reuse attack under the kth code randomization, he should figure out code address space layouts of the kth code randomization in IoT devices and complete code-reuse attack within R_k time. Thus, R_k>τ should hold for a code-reuse attack to be successfully done between the (k−1)th and the kth code randomizations.

As a result, the probability that code-reuse attack succeeds between the first and the second code randomizations given that it fails in the first occurrence of code randomization is calculated as:

$$\begin{aligned} P(R_2 > \tau | R_1 \le \tau) &= \frac{P((R_2 > \tau) \cap (R_1 \le \tau)}{P(R_1 \le \tau)} \\ &= P(R_2 > \tau) \\ &= e^{-\alpha\tau} \end{aligned} \quad (1)$$

By generalizing the above equation to the case of R_k , we have

$$\begin{aligned} p &= P(R_k > \tau | (R_{k-1} \le \tau) \cap \cdots \cap (R_1 \le \tau)) \\ &= e^{-\alpha\tau} \end{aligned} \quad (2)$$

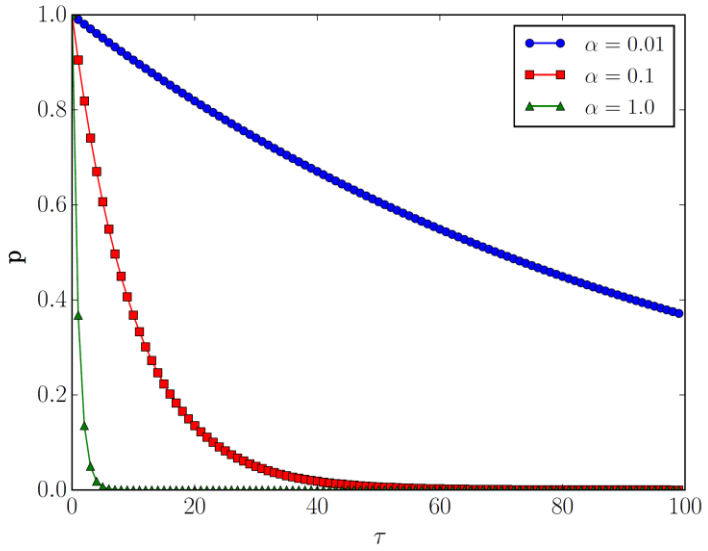

Figure 1. How poisson process parameter α and code-reuse attack time τ impact on probability *p*. Note that τ is represented in simulation seconds.

As shown in Figure 1, we observe that an increase in τ contributes to a decrease in the probability *p* that code-reuse attack succeeds between the (*k*−1)th and the *k*th code randomizations given that it fails in all previous inter-occurrence times of code randomizations. Furthermore, given a value of τ, we see that a decay in α results in a rise in *p*. From these observations, we perceive that the shorter the code-reuse attack time with the higher inter-occurrence time of code randomizations, the higher the likelihood that code-reuse attack will succeed in inter-occurrence time of code randomizations.

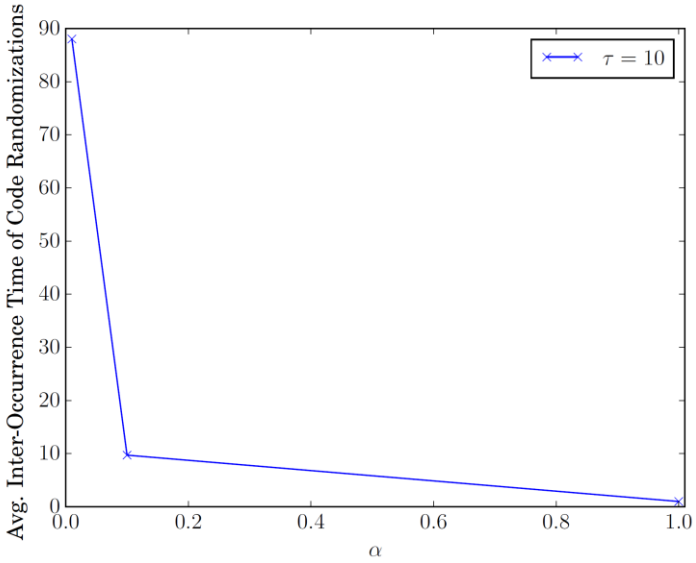

Figure 2. How poisson process parameter α affects an average inter-occurrence time of code randomizations when τ=10 simulation seconds. Note that average inter-occurrence time is represented in simulation seconds.

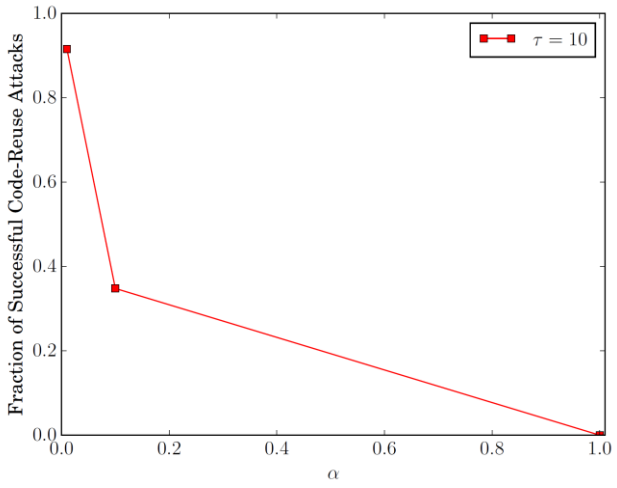

Figure 3. How the fraction of successful code-reuse attacks (i.e. the case that an inter-occurrence time is greater than τ) over all inter-occurrence times of code randomizations is varied in accordance with poisson process parameter α when τ = 10 simulation seconds.

To evaluate the above analysis, we perform a simulation in 1000 seconds through a simple program. In this simulation, we set code-reuse attack time τ to 10 simulation seconds. Since the inter-occurrence times of code randomizations follow the exponential distribution, the inter-occurrence time of code randomizations is calculated as $-\frac{\ln(A)}{\alpha}$, where A ($0 \leq A < 1$) is uniform random variate. We also set α to 0.01, 0.1, 1.0. Note that we repeat the simulation 1000 times and describe the average results of 1000 runs. The number of times of code randomization occurrence is measured as 10.848, 102.987, 1004.94 on an average in the 1000 simulation seconds when α = 0.01, 0.1, 1.0, respectively. Figure 2 shows the interactions between α and average inter-occurrence time of code randomizations. When τ = 10 simulation seconds, we see that an increase in α contributes to a decrease in average inter-occurrence time of code randomizations. As shown in Figure 3, we discern that the fraction of successful code-reuse attacks decreases as α increases. This implies that the shorter inter-occurrence times of code randomizations take effect in diminish of successful code-reuse attacks. In particular, we notice that the fraction of successful code-reuse attacks is computed as 91.586% and 0.003% on an average when τ = 10 simulation seconds and the number of times of code randomization occurrence is 10.848 and 1004.94 on an average in 1000 simulation seconds, respectively.

3. Conclusion

In this paper, we propose a quantitative analysis on the interactions between code randomization defenses and code-reuse attacks in IoT, and the simulation results on that analysis. As a future work, we will plan to evaluate the proposed analysis on the real IoT testbed.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2016R1C1B1014126).

References

[1] T. Bletsch, X. Jiang, and V. Fresh. Mitigating Code-Reuse Attacks with Control-Flow Locking. In *ACSAC*, 2011.

[2] S. Crane, C. Liebchen, A. Homescu, L. Davi, P. Larsen, A.-R. Sadeghi, S. Brunthaler, M. Franz. Readactor: Practical Code Randomization Resilient to Memory Disclosure. In *IEEE S&P*, 2015.

[3] L. Davi, C. Liebchen, A-R. Sadeghi, K. Z. Snow, and F. Monrose. Isomeron: Code Randomization Resilient to (Just-In-Time) Return-Oriented Programming. In *NDSS*, 2015.

[4] J. Habibi, A. Panicker, A. Gupta, and E. Bertino. DisARM: Mitigating Buffer Overflow Attacks on Embedded Devices. In *CERIAS Tech Report*, 2015-15, 2015.

[5] R. Roemer, E. Buchanan, H. Shacham, and S. Savage. Return-oriented programming: Systems, languages, and applications. In *ACM Transactions on Information and System Security*, 15, 1 (Mar. 2012), 2:1-2:34.