Review of “Methodologies for Quantifying (Re-)randomization Security and Timing under JIT-ROP”

Xin Liu
University of Virginia
Charlottesville, USA
xl9jv@virginia.edu

ABSTRACT

The paper is a brief review on the paper “Methodologies for Quantifying (Re-)randomization Security and Timing under JIT-ROP” [2]. The review will give a brief summary of the paper followed by the advantages and disadvantages of the paper. Finally, the review will post some questions relating to the paper and conclude with potential discussions.

ACM Reference Format:
Xin Liu. 2018. Review of “Methodologies for Quantifying (Re-)randomization Security and Timing under JIT-ROP”. In *Woodstock ’18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 SUMMARY

Since the impact of code (re-)randomization on code reuse attacks have not been studied, the paper [2] conducts a measurement study on the effectiveness of code randomization schemes. They measure the availability, quality, and their Turing-complete (TC) expressiveness of gadget. Furthermore, they use TC to obtain the upper bound of re-randomization intervals. The experimental results conclude that the upper bound range is between 1.5s and 3.5s. In addition, they find that the location of pointer leakage does not impact gadget availability but it impacts the speed of attackers to find gadgets. Finally, they also discover that single-round randomization in instruction level will impact the gadget finding techniques. In conclusion, the paper conducts many methodologies to analyze the fine-grained address space lay- out randomization (ASLR) protection under the JIT-ROP (Just-in-time return-oriented programming) threat model. There are two different methods, one is for measuring types and quality of gadgets, and another is for computing the upper bound of re-randomization to help defenders make decisions of configurations.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Woodstock ’18, June 03–05, 2018, Woodstock, NY
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

2 ADVANTAGES

2.1 Measurement methodology to evaluate ASLR’s effectiveness

Directly launching exploits is not useful due to low scalability, low reproducibility, and confusion of the judgement of security. The paper designs a novel mechanism to compare results under multiple ASLR conditions. Furthermore, it can perform the JIT-ROP attacks in a scalable fashion. To quantify the impact, the method counts the number of available gadgets under the JIT-ROP. To quantify the quality of gadget chains, they computing the register corruption rate. To quantify the difficulty of accessing privileged operations, they compute the number of system gadgets and count *libc* pointers. They are really helpful to evaluate the impact of ASLR.

2.2 Measure the quality of individual gadgets

They give a detailed analysis of register corruption for each gadget to measure the quality of individual gadgets. Furthermore, they measure the register corruption rate by analyzing how the core instruction of a gadget can get modified. To consider multiple core instructions, they utilize the core instruction that is closest to the ret instruction. This is one of the most interesting steps.

3 DISADVANTAGES

3.1 Types of gadgets for evaluation

They use the Turing-complete, priority, and MOV TC gadget sets for the evaluation because they can precisely identify those gadgets in the extracted 21 types of gadgets. But for those gadgets that are not included for evaluations, they also impact the process of experiments, the paper ignores their impacts here and they should be considered.

3.2 Methodology for re-randomization experiments

They take 100 consecutive address space snapshots from an application/library re-randomized by Shuffler and manually analyze the address space snapshots to evaluate the impact of re-randomization. There should be a more efficient method to analyze the space snapshots.

4 QUESTIONS

Questions regarding the technical details.

- What is the definition of gadget in this paper? What is their function? What does the quality of a gadget chain mean? We need a better understanding of gadgets.

- Shorter intervals (e.g., millisecond-level) incur runtime overhead, why not to compute a low bound of re-randomization intervals? Do they ignore it or it will not impact a lot?
- Since they can precisely identify those gadgets, such as Turing-complete, priority, and MOV TC gadget sets, they only use them. What about other types? If they only can be identifies roughly, how to consider them in the evaluation?
- For the measurement of vulnerable pointers in a stack/heap/data-segment, they identify the number of unique *libc* pointers. Why *libc* pointers? Are there any other pointers? They do not exploit vulnerabilities to leak *libc* pointers, but how do they know the address mapping of *libc*? What about the linear scanning?
- In the case of multiple core instructions of a gadget type, they consider the core instruction that is closest to the ret instruction. Why ret instructions? There should be more explanations.

5 DISCUSSIONS

5.1 Evaluation for fine-grained randomization

Common randomizations use entropy to measure the effectiveness of hindering code-reuse attacks. And JIT-ROP attacks leverage on code connectivity to discover code page. Designing an entropy metric to compute the degree of code isolation and code connectivity is a challenge.

5.2 Availability of gadgets

They design the experiments based on the availability of various kinds of gadgets. However, in reality, attackers need to conduct a series of operations including finding a vulnerability or leaking memory for the actual invocations of gadgets. In this paper ,they assume an attacker has already overcome the initial obstacles, but what will happen if we do not know the practical strength of attacks. There should be more analysis of gadget availability.

6 SOURCE TEMPLATE

Find it on the ACM website [1].

REFERENCES

[1] ACM. 2020. ACM Master Article Template. https://www.acm.org/publications/proceedings-template.

[2] Salman Ahmed, Ya Xiao, Gang Tan, Kevin Snow, Fabian Monrose, Danfeng, and Yao. 2020. Methodologies for Quantifying (Re-)randomization Security and Timing under JIT-ROP. arXiv:1910.03034 [cs.CR]