ITRG Mini-proposal

Extending Code Reuse Analysis Framework for Trusted and Effective Defense with ARM Instruction Set

Jie Zhou
Department of Computer Science, University of Rochester

1 Project Summary

Computer security has always been an important issue for both professional computer researchers and regular computer users. For computer security, memory safety is one of the most significant areas. And for memory safety, code reuse attacks like return-to-libc attacks [1], Return-oriented programming (ROP) [2], None-control data attacks [3], etc. have shown their power in controlling the whole target operating system by exploiting memory safety errors such as buffer overflow bugs in the past two decades. To prevent code reuse attacks, a large number of methods have been proposed and carried out for operating systems. Along with the development of defense policies, however, are smarter attack techniques that circumvent the defense policies. Then later new defense strategies come up to thwart the aforementioned "smarter" attacks. The war seems eternal. Nowadays as the mobile devices (mainly smartphones and tablets) play a more and more important role in people's life, the security of them has become a major issue. Most mobile devices today are running on Unix-based operating systems like iOS and Android, which means they share the basic mechanism and principles with systems (such as Linux and MacOS) of numerous traditional desktop computers. It also means that code reuse attack techniques working on desktop computing devices can be applied to mobile devices as well. That is why security of mobile devices has been as concerning an issue as desktop devices.

As the "Eternal War in Memory" [4] goes on, although ingenious defense policies against code reuse attack have been designed and applied one after another, they are ad hoc. There had been no general and systematic approaches that could verify the efficacy of any given defense policy against code reuse attacks until Code Reuse Analysis Framework for Trusted and Effective Defense (CRAFTED) [5] was proposed. Given a defense policy for code reuse attacks, a target program with potential vulnerabilities, a malicious computation template, CRAFTED aims to do static analysis to determine the efficacy of the policy. Thus, for any new defense strategy, CRAFTED can determine if it is robust against a certain malicious computation model; otherwise there is a chance that the strategy works fine based on its designer's experiments but are defeated by a cleverer attack scheme later. Currently, CRAFTED is still under construction, and it can only analyze x86 target code. Today the mainstream architectures used in mobile devices such as iOS and Android are from ARM family. Therefore, the principal investigator proposes to extend CRAFTED infrastructure with

support for ARM instruction set so that CRAFTED can analyze and evaluate the efficacy of defense policies used for mobile devices.

2 Intellectual Merit

The proposed research intends to enhance CRAFTED with static analysis on programs and defense policies for mobile devices with ARM processors. The ARM instruction set will be explored in great details so that its security mechanism will be further understood. It will also help code reuse attack researchers test and perfect their design and implementation of defense policies for ARM devices.

3 Broader Impacts

The proposed work can facilitate the process of analysis and improvement of defense policies against code reuse attacks on devices equipped with ARM chips. The benefit for society is obvious: till March 2017 there were over 700 million iPhones in use [6], and till May 2017 there were over 2 billion active Android devices [7]); thus, enhancing the security for mobile devices indirectly by boosting the development of security policies would be valuable to billions of people. Besides, CRAFTED is based on LLVM compiler infrastructure [8]. The investigators will release the source code of the project to the public under open-source licenses, which will not only help enrich the LLVM community, but also help other security researchers better understand code reuse attacks and devise superior defense tools.

4 Research Plan

The research will be conducted in three phases.

In the first phase, the investigators will learn and understand CRAFTED thoroughly, because the proposed work is based on it. And at the same time the investigators will also need to have a good grasp of ARM instruction set, which is also fundamental to the work.

The second phase is to implement ARM version of all the tools based on x86 versions in current CRAFTED: target program analysis, malicious computation template analysis, defense policy efficacy verifier, etc. The algorithms and code styles should be consistent with their corresponding x86 parts.

The last phase is to conduct extensive tests with real-world programs like popular applications on iOS and Android phones. In addition to correctness, overheads of the newly implemented work will also be taken into consideration.

Reference

[1] Solar Designer. return-to-libc attack, August 1997. http://www.securityfocus.com/archive/1/7480.
[2] Ryan Roemer, Erik Buchanan, Hovav Shacham, and Stefan Savage. Return-oriented programming: Systems, languages, and applications. ACM Transactions on Information Systems Security, 15(1):2:1– 2:34, March 2012.
[3] Shuo Chen, Jun Xu, Emre C. Sezer, Prachi Gauriar, and Ravishankar K. Iyer. Non-control-data attacks are realistic threats. In 14th USENIX Security Symposium, pages 177–192, August 2004.
[4] László Szekeres, Mathias Payer, Tao Wei, and Dawn Song. SoK: Eternal war in memory. IEEE Symposium on Security and Privacy, 2013.
[5] Ethan Johnson, Tianqin Zhao, John Criswell. Poster: CRAFTED: Code Reuse Analysis for Trusted and Effective Defense. IEEE Symposium on Security and Privacy, 2017.
[6] Fortune Magazine. http://fortune.com/2017/03/06/apple-iphone-use-worldwide/
[7] androidcentral.com. https://www.androidcentral.com/there-are-over-2-billion-active-android-devices-today
[8] Chris Lattner and Vikram Adve. LLVM: A compilation framework for lifelong program analysis and transformation. In Proceedings of the Conference on Code Generation and Optimization, pages 75–88, San Jose, CA, USA, Mar 2004.