Reprogramming embedded systems using Return Oriented Programming

Sam Mitchell and Nathanael Weidler
Deptartment of Electrical and Computer Engineering
Utah Stat University
Logan, Utah 84322
e-mail: samuel.alan.mitchell@gmail.com, NWeidler@gmail.com

***Abstract*—This paper describes the implementation of high-throughput password cracking devices. We consider architecture-aware implementations of password crackers on FPGA and X86 architectures. An analysis of speed and cost efficacy is included. This paper describes the theory and implementation of a return-oriented programming attack on a TM4C123GH6PM microcontroller. We describe the gadget searching process, as well as the injection method.**

***Index Terms*—Return Oriented Programming, Security, ARM, Harvard architecture.**

I. Introduction

Single-purpose embedded devices such as voting machines or vehicle guidance controllers are generally considered to be secure machines. Previous work has shown that Return Oriented Programming (ROP) methods can be used to clear and reset Harvard architecture-devices.

ROP is a type of buffer-overflow attack that modifies the return address of the existing program, causing the program to execute existing code. In x86 architectures, ROP attacks generally jump into libc to control the behavior of the compromised program. Harvard architectures use similar techniques — manufacturer-provided peripheral driver libraries provide sufficient code-space to implement a devastating ROP attack.

A. Structure of paper

The organization is as follows: in Section II, the development of a software-based MD5 password cracking device is presented and analyzed. Section III contains the focus and development of a hardware-based MD5 password cracking device. Conclusions are presented in Section IV.

II. System design

III. Required assembly

IV. Gadgets

V. ROP design

VI. Implementation and results

VII. Conclusion

This paper considers the efficient computation of passwords. Multiple methods to increase hashing throughput are presented. It is shown that hardware implementation of a password cracker provides more throughput per unit dollar than a software implementation. Future research will address the efficacy of different architectures in password computation.

References