Project Proposal

Jonathan Rudman

Problem Statement

Microcontroller units (MCUs) are found embedded in many commercially available devices, performing many functions as part of a greater device. In order to protect intellectual property, avoid cloning and prevent the discovery of software vulnerabilities, the firmware of a microcontroller is protected by disabling debugging and read/write of internal memory.

The different implementations of these protections have various weaknesses that can be exploited through implementation attacks, and it is important for manufacturers of MCUs to reduce the effectiveness or increase the difficulty of these attacks. One such weakness which is common among MCUs is voltage glitching: the act of raising or lowering the operating voltage for a fraction of a second to cause instructions to be skipped or incorrectly executed. This can be used to gain access to a debug interface and enable reading the firmware to allow the attacker to modify existing proprietary firmware or search for software vulnerabilities.

I will be attempting to employ implementation attacks such as voltage glitching to dump the protected firmware of microcontrollers. This may be informed by binary analysis of an available bootloader source and software vulnerabilities to dump protected internal memory.

Related Work

Roth (2021) has shown how the nRF52832 in Apple AirTags can be glitched to skip its code readout protection (CRP) check, which I aim to replicate at the start of this project. Roth also cites earlier work from a blog, LimitedResults (2020), which focuses on a similar system-on-a-chip (SoC), the nRF52840, and presents information from the nRF52 family in the context of voltage glitching and CRP.

Van den Herrewegen *et al.* (2020) describe voltage glitching in conjunction with binary analysis—a grey-box approach—on embedded bootloaders. Although the nRF52832—the SoC I intend to attack first—has a CRP check before the bootloader (which is stored on flash memory), Van Den Herrewegen et al. outline voltage glitches on three different MCUs. In the aforementioned paper the GIAnT, Oswald (no date), is also used for the attacks, which is a device that I plan to use and develop further.

Challenges

There are a number of challenges that this project will present:

- I've never done this much with electronics before. I did physics A level but I've never attempted power analysis on integrated circuits and I've only flashed firmware based on specific instructions.
- Reducing the glitch parameter space could involve binary analysis and/or power monitoring; the alternative is to use brute force, which is time consuming.
- There are many microcontrollers for which voltage glitching hasn't been attempted or successful yet. As a "stretch" goal I'd like to attempt to glitch an MCU without any form of guide. For example, I'm not aware of a successful voltage glitch attack on the Texas Instruments CC2541.
- Automating the parameter space can be done in various different ways and can make voltage glitching more accessible, leading to the potential for more complex attacks. Extending and developing software to do this more effectively is a challenge that intrigues me.

Planned Approach

I plan to connect a device for running voltage glitching attacks—such as the Pico Debug'n'Dump (Roth (no date)) or GIAnT—to microcontrollers to gain access to a debugging interface. This will likely be done with some supplementary binary analysis, depending on whether the bootloader is on ROM, to determine how many cycles into the boot process the instructions for checking debug and read/write flags are. I also plan to extend GIAnT with support for more MCUs and research the possibility of reducing the number of glitches required to find the correct glitch offset, width and voltage for a successful attack.

Milestones

Semester One

18th October 2021

Successfully replicate the voltage glitch attack on the nRF52832. At the time of writing (15/10/2021) I have set up the hardware with an nRF52832 development board, Pico Debug'n'Dump and an ST-LINK V2 serial debugger and dumped firmware from the nRF52832 without code readout protection enabled.

8th Nov 2021 (Project inspection)

Should have made contributions to GIAnT to add nRF52832 support. Be able to demonstrate/show results from a successful voltage glitch attack. If the previous deadline is met, show contributions to GIAnT and be able to answer questions.

19th Nov 2021

Should have tested and compiled a list of success rates for glitching with various parameters on the nRF52832 and made notes on findings and lessons learned from multiple glitch attempts.

10th Dec 2021

Should have begun researching and reading the datasheet for an MCU which hasn't experienced a documented glitch (e.g. CC2541).

Semester Two

21st Jan 2022

Should have successfully glitched the chosen MCU from the last deadline and made notes on findings.

25th Feb 2022

Should have investigated ways to automate—or adjust current automation methods for—voltage glitching effectively with MCUs which haven't yet been successfully glitched. Should have used findings to propose mitigations for voltage glitching and any other techniques used throughout the project.

11th Apr 2022 (Final submission)

Should have created the report for final submission.

Degree Apprenticeship Specialisation

This project will focus on developing skills and knowledge listed under the "Software Engineering Specialist" section of the Digital and Technology Solutions Professional degree apprenticeship standard outlined by the Institute for Apprenticeships.

Skills

I plan to meet the following skills requirements:

Knowledge

I plan to meet the following technical knowledge requirements:

Reference List

LimitedResults (2020) *nRF52 debug resurrection (APPROTECT bypass) part 1. LimitedResults*. Available at: https://limitedresults.com/2020/06/nrf52-debug-resurrection-approtect-bypass/ (Accessed: 15 October 2021).

Oswald, D. (no date) *Giant-revB: The GIAnT fault injection board in the new revision. GitHub*. Available at: https://github.com/david-oswald/giant-revB (Accessed: 15 October 2021).

Roth, T. (2021) *How the apple AirTags were hacked*. Available at: https://www.youtube.com/watch?v=_E0PWQvW-14 (Accessed: 15 October 2021).

Roth, T. (no date) *Pico debug'n'Dump. Pico debug'n'Dump*. Available at: https://pdnd.stacksmashing.net/ (Accessed: 15 October 2021).

Van den Herrewegen, J. *et al.* (2020) 'Fill your boots: Enhanced embedded bootloader exploits via fault injection and binary analysis', *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 56–81. doi:10.46586/tches.v2021.i1.56-81.