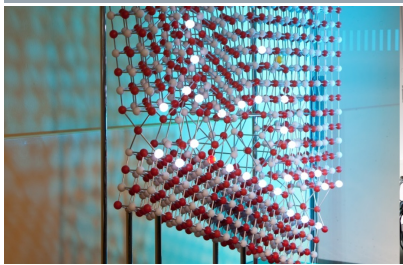
FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

Rehosting Basics

Rob
PWN
June 4, 2022



Bilder: TF / Malter

Outline

Outline for Introduction

Analysis Techniques

Analysis Techniques:

Dynamic

- debugging
- fuzzing

Static

- reverse engineering
- source code audit

Analysis Techniques

Analysis Techniques:

Dynamic

- debugging
- fuzzing

Static

- reverse engineering
- source code audit

Hardware

Dynamic

- no debugger access
- expensive with real devices

Static

- time consuming
- no source code

Rehosting

Definition

- Decouple hardware from software.
- **Emulate hardware for extracted firmware.**

Rehosting gives us a platform for analyzing firmwares.

Rehosting

Definition

- Decouple hardware from software.
- **Emulate hardware for extracted firmware.**

Rehosting gives us a platform for analyzing firmwares.

Challenges

- Hardware is complex.
- No emulator can emulate whole device.
- software stacks are complex.

Rehosting

Decoupling a system's firmware from its underlying hardware to move—or rehost— the software into a virtual environment designed to run that firmware - Fasanoe, Ballo, Muench, et al. [1]

Terminology

Virtual Execution Engine
: Mechanism for interpreting Instructions

Virtual Engine
: Provides the runtime for the VXE

Hardware Emulation System
: VE+VXE which emulate a specific embedded device

Rehosted Emulation System
: A HES in which a rehosted firmware can be executed

Related Work

At present, the process of rehosting is more alchemy than chemistry— opaque, unrepeatable, and prone to failure. - Fasanoe, Ballo, Muench, et al. [1]

Related Work

At present, the process of rehosting is more alchemy than chemistry— opaque, unrepeatable, and prone to failure. - Fasanoe, Ballo, Muench, et al. [1]

SoK Rehosting

Criteria:

- Introspection
- Correctness/*Fidelity*
- Scalability
- Disposability

Related Work

At present, the process of rehosting is more alchemy than chemistry— opaque, unrepeatable, and prone to failure. - Fasanoe, Ballo, Muench, et al. [1]

SoK Rehosting

Criteria:

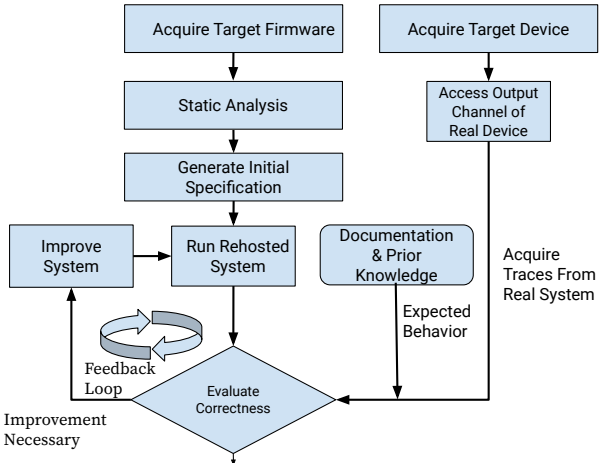
- Introspection
- Correctness/*Fidelity*
- Scalability
- Disposability

PartEMU

Core Ideas:

- Only rehost the necessary part of the Firmware
- Emulate well defined interfaces!
- Reuse for complex components!

Approach


Acquire Target Firmware
Acquire Target Device
Static Analysis
Access Output Channel of Real Device
Generate Initial Specification
Improve System
Run Rehosted System
Documentation & Prior Knowledge
Acquire Traces From Real System
Expected Behavior
Feedback Loop
Evaluate Correctness
Improvement Necessary

Avatar2

Core Ideas

- Orchestrate the execution of multiple targets
- Separation between execution and memory
- Build for Hardware-in-the-Loop solutions

Avatar2

Core Ideas

- Orchestrate the execution of multiple targets
- Separation between execution and memory
- Build for Hardware-in-the-Loop solutions

Configurable Machine

- QEMU Board for System Mode
- Configuration according to JSON file
- supports ARM32, MIPS, AARCH64

Avatar2 tidbits

Positives

- Can handle ArmV8 including every **EL**
- offers powerfull synchronization primitives
- easyish to extend

Negatives

- every modification is an rpc call into the GdbStub
- slow

Unicorn Engine

Core Ideas

- Just the execution Engine of QEMU (TCG)
- **You build the whole board**

Unicorn Engine

Core Ideas

- Just the execution Engine of QEMU (TCG)
- **You build the whole board**

Hooks

- We need **HOOKS** to handle exceptions
- can be defined for multiple events

Unicorn Engine tidbits

Positives

- easier to setup
- **You build the whole board**

Unicorn Engine tidbits

Positives

- easier to setup
- **You build the whole board**

Negatives

- **HOOK** primitive is way less powerfull
- the emulation assumes to run in kernel mode

Today

Circuit Playground

- We will use the blinky_basic
- we will use only the binary

Today

Circuit Playground

- We will use the blinky_basic
- we will use only the binary

the plan!

- find the mappings
- build an HES
- modify the execution flow accordingly
- optionally: try to make the red LED blink :)

Thanks for listening!
Any questions?

Outline for References

Introduction

References I

[1] A. Fasanoe, T. Ballo, M. Muench, *et al.*, "Preprint: Sok: Enabling security analyses of embedded systems via rehosting," in *The 16th ACM ASIA Conference on Computer and Communications Security)*, Jun. 2021.

[2] G. Wen, "El3 tour: Get the ultimate privilege of android phone,", 2019. [Online]. Available: https://downloads.immunityinc.com/infiltrate2019-slidepacks/guanxing-wen-el3-tour-get-the-ultimate-privilege-of-android-phone/infiltrate.pdf.

[3] M. Bley, "Exploiting the arm trusted firmware. a case study on huawei mobile devices,", 2020.

[4] L. Harrison, H. Vijayakumar, R. Padhye, K. Sen, and M. Grace, "PARTEMU: Enabling dynamic analysis of real-world trustzone software using emulation," in *29th USENIX Security Symposium (USENIX Security 20)*, USENIX Association, Aug. 2020, pp. 789–806, ISBN: 978-1-939133-17-5. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/harrison.