User-Controlled Hardware Security Anchors: Evaluation And Designs

Mark Ryan, Flavio Garcia, David Oswald and Eduard Marin

2nd RISE Annual Conference

London, 21/11/2019

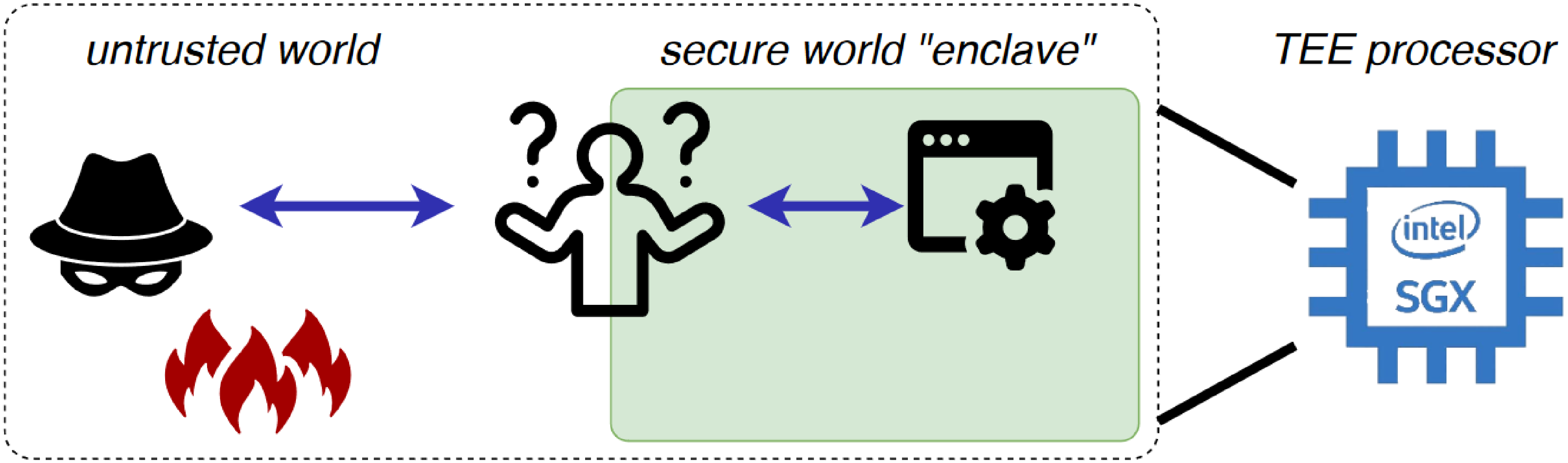
UNIVERSITY OF BIRMINGHAM

Attacks

Lots of attacks in recent years…

All these attacks exploit microarchitectural bugs or side channels at the hardware level

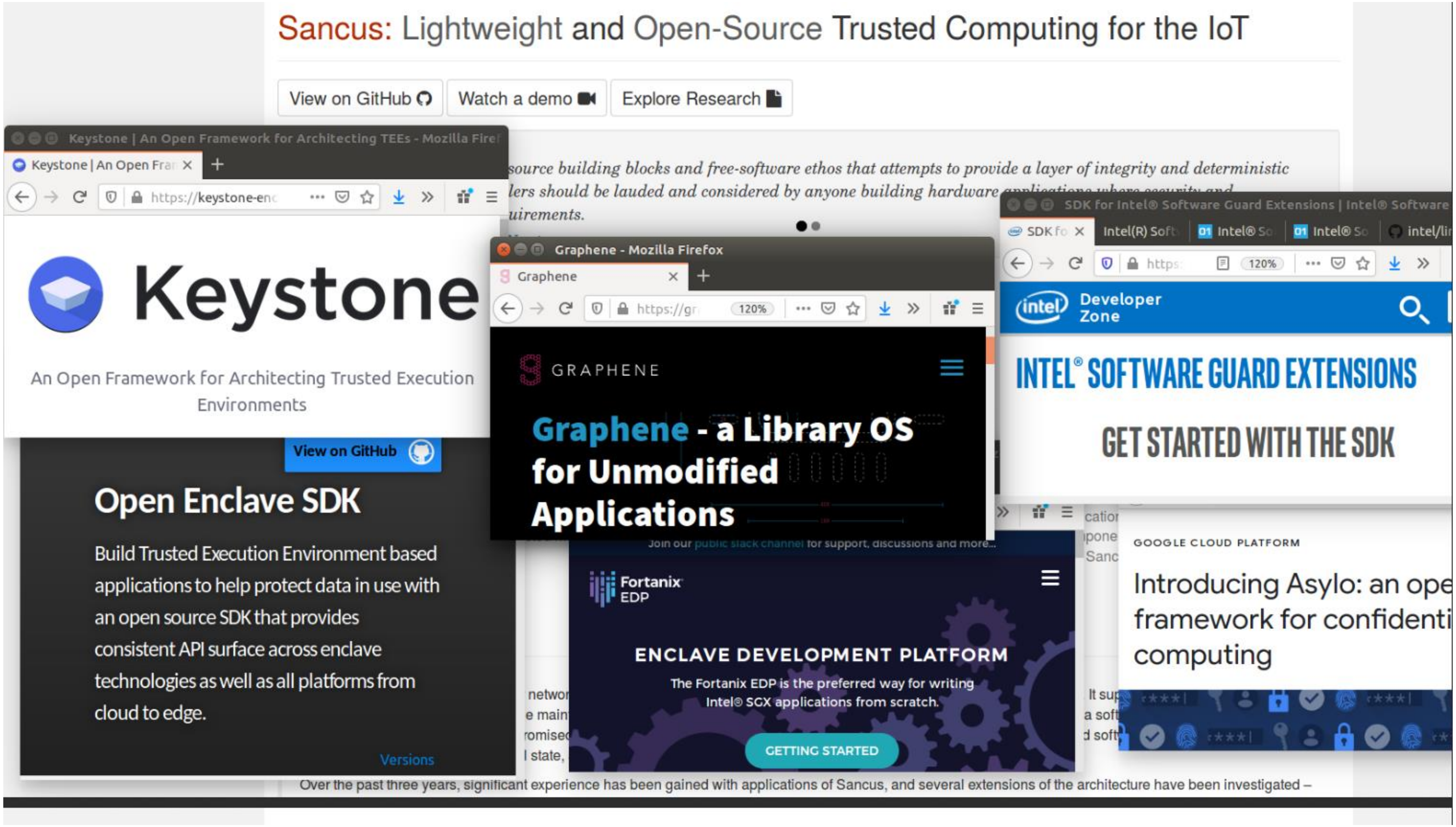
Enclave shielding runtimes



TEE promise: enclave == “secure oasis” in a hostile environment

but application writers and compilers are largely unaware of isolation boundaries!

<u>Trusted shielding runtime transparently acts as a secure bridge on enclave entry/exit</u>


Sancus: Lightweight and Open-Source Trusted Computing for the IoT
View on GitHub
Watch a demo
Explore Research
Keystone | An Open Framework for Architecting TEEs - Mozilla Firef
Keystone
An Open Framework for Architecting Trusted Execution Environments
View on GitHub
Open Enclave SDK
Build Trusted Execution Environment based applications to help protect data in use with an open source SDK that provides consistent API surface across enclave technologies as well as all platforms from cloud to edge.
Versions
Graphene - Mozilla Firefox
GRAPHENE
Graphene - a Library OS for Unmodified Applications
SDK for Intel® Software Guard Extensions | Intel® Software
Developer Zone
INTEL® SOFTWARE GUARD EXTENSIONS
GET STARTED WITH THE SDK
Fortanix EDP
ENCLAVE DEVELOPMENT PLATFORM
The Fortanix EDP is the preferred way for writing Intel® SGX applications from scratch.
GETTING STARTED
GOOGLE CLOUD PLATFORM
Introducing Asylo: an ope framework for confidenti computing
Over the past three years, significant experience has been gained with applications of Sancus, and several extensions of the architecture have been investigated –

Our recent paper at CCS'19

A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes

Jo Van Bulck
imec-DistriNet, KU Leuven
jo.vanbulck@cs.kuleuven.be

David Oswald
The University of Birmingham, UK
d.f.oswald@cs.bham.ac.uk

Eduard Marin
The University of Birmingham, UK
e.marin@cs.bham.ac.uk

Abdulla Aldoseri
The University of Birmingham, UK
axa1170@student.bham.ac.uk

Flavio D. Garcia
The University of Birmingham, UK
f.garcia@cs.bham.ac.uk

Frank Piessens
imec-DistriNet, KU Leuven
frank.piessens@cs.kuleuven.be

ABSTRACT

This paper analyzes the vulnerability space arising in Trusted Execution Environments (TEEs) when interfacing a trusted enclave application with untrusted, potentially malicious code. Considerable research and industry effort has gone into developing TEE runtime libraries with the purpose of transparently *shielding* enclave application code from an adversarial environment. However, our analysis reveals that shielding requirements are generally not well-understood in real-world TEE runtime implementations. We expose several sanitization vulnerabilities at the level of the Application Binary Interface (ABI) and the Application Programming Interface (API) that can lead to exploitable memory safety and side-channel vulnerabilities in the compiled enclave. Mitigation of these vulnerabilities is not as simple as ensuring that pointers are outside enclave memory. In fact, we demonstrate that state-of-the-art mitigation techniques such as Intel's `edger8r`, Microsoft's "deep copy marshalling", or even memory-safe languages like Rust fail to fully eliminate this attack surface. Our analysis reveals 35 enclave interface sanitization vulnerabilities in 8 major open-source shielding frameworks for Intel SGX, RISC-V, and Sancus TEEs. We practically exploit these vulnerabilities in several attack scenarios to leak secret keys from the enclave or enable remote code reuse. We have responsibly disclosed our findings, leading to 5 designated CVE records and numerous security patches in the vulnerable open-source projects, including the Intel SGX-SDK, Microsoft Open Enclave, Google Asylo, and the Rust compiler.

London, UK. ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3319535.3363206

1 INTRODUCTION

Minimization of the Trusted Computing Base (TCB) has always been one of the key principles underlying the field of computer security. With an ongoing stream of vulnerabilities in mainstream operating system and privileged hypervisor software layers, Trusted Execution Environments (TEEs) [28] have been developed as a promising new security paradigm to establish strong hardware-backed security guarantees. TEEs such as Intel SGX [8], ARM TrustZone [34], RISC-V Keystone [21], or Sancus [32] realize isolation and attestation of secure application compartments, called *enclaves*. Essentially, TEEs enforce a dual-world view, where even compromised or malicious system software in the normal world cannot gain access to the memory space of enclaves running in an isolated secure world on the same processor. This property allows for drastic TCB reduction: only the code running in the secure world needs to be trusted for enclaved computation results. Nevertheless, TEEs merely offer a relatively coarse-grained memory isolation primitive at the hardware level, leaving it up to the enclave developer to maintain useful security properties at the software level. This can become particularly complex when dealing with interactions between the untrusted host OS and the secure enclave, e.g., sending or receiving data to or from the enclave. For this rea-

What did we find?

Tier	Vulnerability \ Runtime	SGX-SDK	OpenEnclave	Graphene	SGX-LKL	Rust-EDP	Asylo	Keystone	Sancus
Tier1 (ABI)	#1 Entry status flags sanitization	★	★	◐	●	◐	●	○	○
	#2 Entry stack pointer restore	○	○	★	●	○	○	○	★
	#3 Exit register leakage	○	○	○	★	○	○	○	○
Tier2 (API)	#4 Missing pointer range check	○	★	★	★	○	●	○	★
	#5 Null-terminated string handling	⯪	★	○	○	○	○	○	○
	#6 Integer overflow in range check	○	○	●	○	●	○	●	●
	#7 Incorrect pointer range check	○	○	●	○	○	●	○	●
	#8 Double fetch untrusted pointer	○	○	●	○	○	○	○	○
	#9 Ocall return value not checked	○	★	★	★	○	●	★	○
	#10 Uninitialized padding leakage	[23]	★	○	●	○	●	★	★

Summary: > 35 enclave interface sanitization vulnerabilities across 8 projects

What did we find?

Responsible disclosure

- Fortanix-EDP => Security patch in the Rust compiler
- Intel SGX => CVE-2018-3626 and CVE-2019-14565
- Microsoft Open Enclave => CVE-2019-0876, CVE-2019-1369 and CVE-2019-1370
- Findings in open-source projects have been acknowledged in Github

All of our attack code is available in Github:

[https://github.com/jovanbulck/0xbadc0de.](https://github.com/jovanbulck/0xbadc0de)

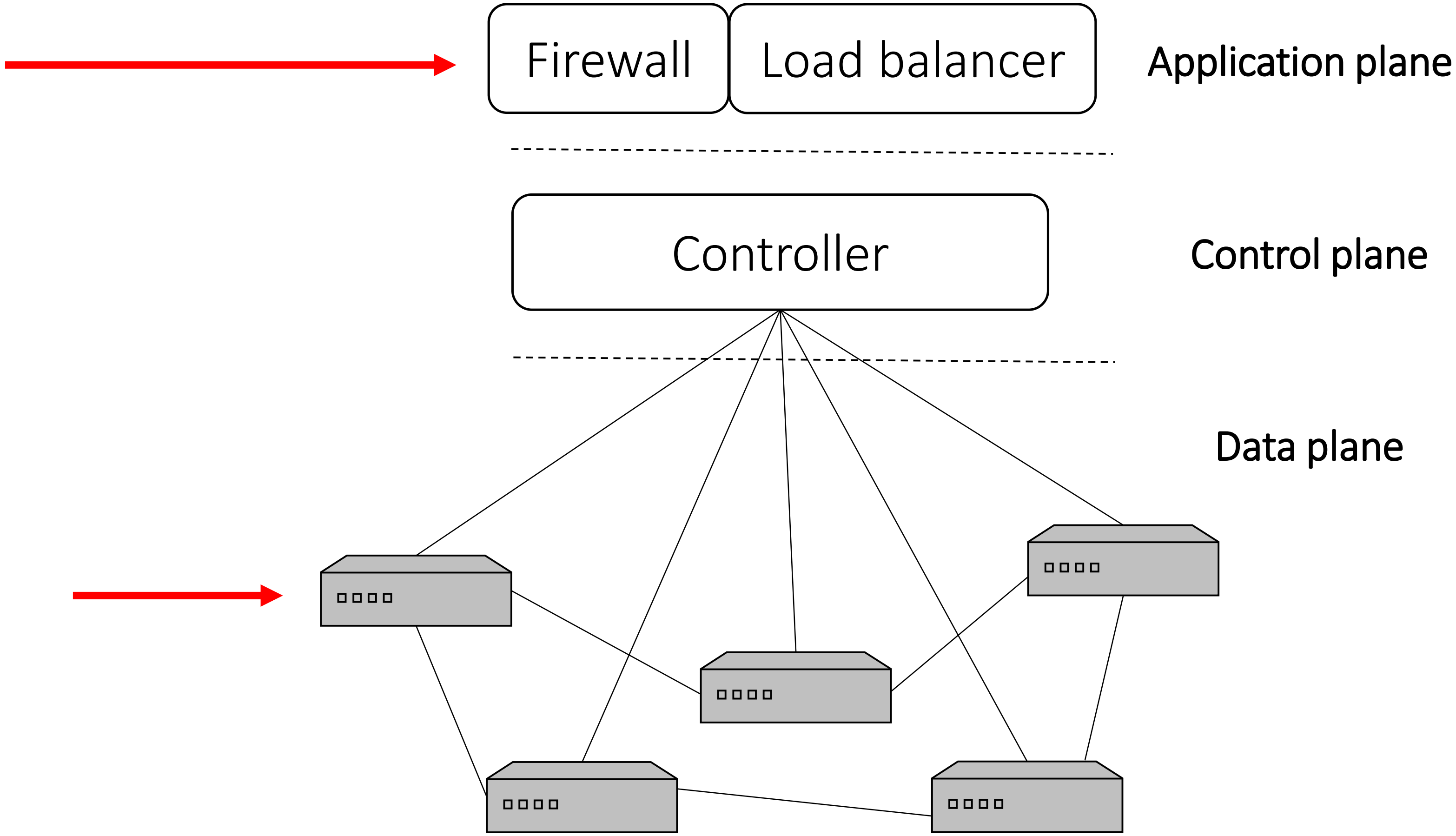
Paper at S&P 2020!!

Defenses

Software defined networks

Different apps written by app developers (App store)

Physical switches can now be divided into several various switches



Improve user authentication

- Use context information (e.g., geographical location)

=> Trusted input from sensors

- Bring your own device (BYOD) scenarios

=> Adversary can have root privileges in the device