Owned or pwned?

no peekin' or tweakin'!

By Richard Zak and Nick Vidal

TABLE OF CONTENTS

Confidential Computing

Confidential Computing

“Confidential Computing protects data in use by performing computation in a hardware-based Trusted Execution Environment. These secure and isolated environments prevent unauthorized access or modification of applications and data while in use, thereby increasing the security assurances for organizations that manage sensitive and regulated data.”

Protections for Data...

At Rest

File encryption on HD/SSD or Cloud Storage

In Use

CPU/Memory (TEEs)

In Transit

HTTPS (Browser ⇔ Server)
End-to-End

Data Protection

- Data Integrity
- Data Confidentiality
- Code Integrity
- Code Confidentiality

Sectors

Defense

Banking/Finance

Healthcare

Government

Customer Data

Telecom

Advantages

No Peekin'

Data & Code Confidentiality

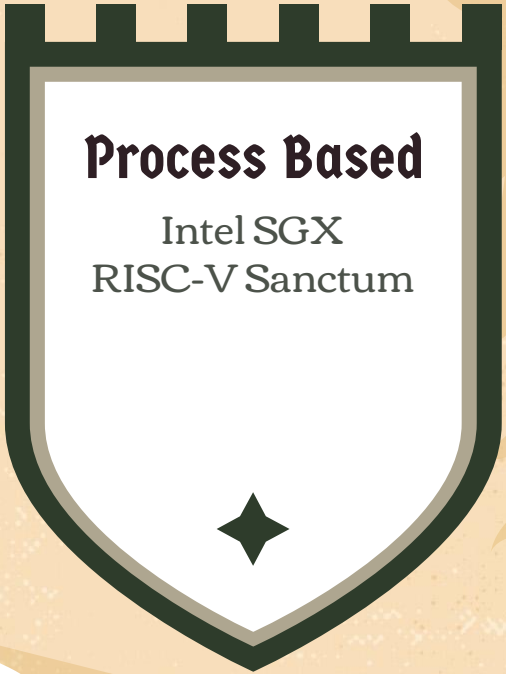
No Tweakin'

Data & Code Integrity

Even if host is compromised
Internal/external threats

Hardware

Architectures



CPUs (early gen)

2012

ARM TrustZone

Secure OS
(Two Worlds)

2015

Intel SGX

Isolated Apps
(Enclave)

2020

AMD SEV

Isolated VMs

2021

IBM PEF

Isolated VMs

CPUs/GPUs (next gen)

2021

AMD SEV-SNP

EPYC Milan

2021

Intel SGX

Xeon Ice Lake

2022

NVIDIA

H100 Hopper

Cloud: next gen general availability

2022

Azure

2022

Google Cloud

Enarx

Enarx

Open Source

Leading framework for running applications in TEEs (Trusted Execution Environments).

Easy Deployment

Provides a run-time TEE based on WebAssembly, allowing developers to deploy apps from various languages

Hardware Neutral

CPU-architecture independent, letting developers deploy the same app binary transparently across multiple targets, different architectures.

WebAssembly

Security

Its sandbox model offers an extra layer of protection, isolating the application from the host.

Performance

It's statically typed and designed to be encoded in a size- and load-time-efficient binary format highly optimized for performance.

Portability

It's designed to be executable efficiently on a variety of operating systems and instruction set architectures, on the Web and off.

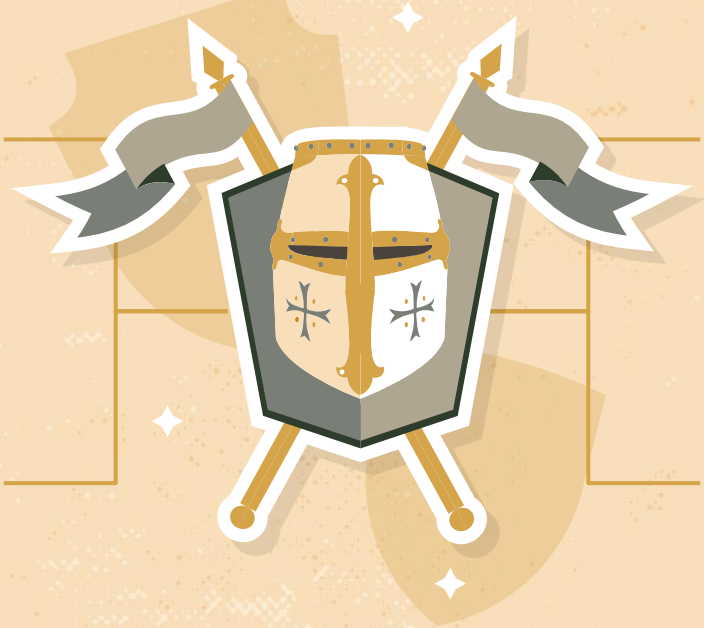
Enarx Architecture

Enarx Keep

Sets up the TEE & runs the app

Steward

Performs attestation & provides crypto cert



Drawbridge

Acts as a repo for WebAssembly apps

Host

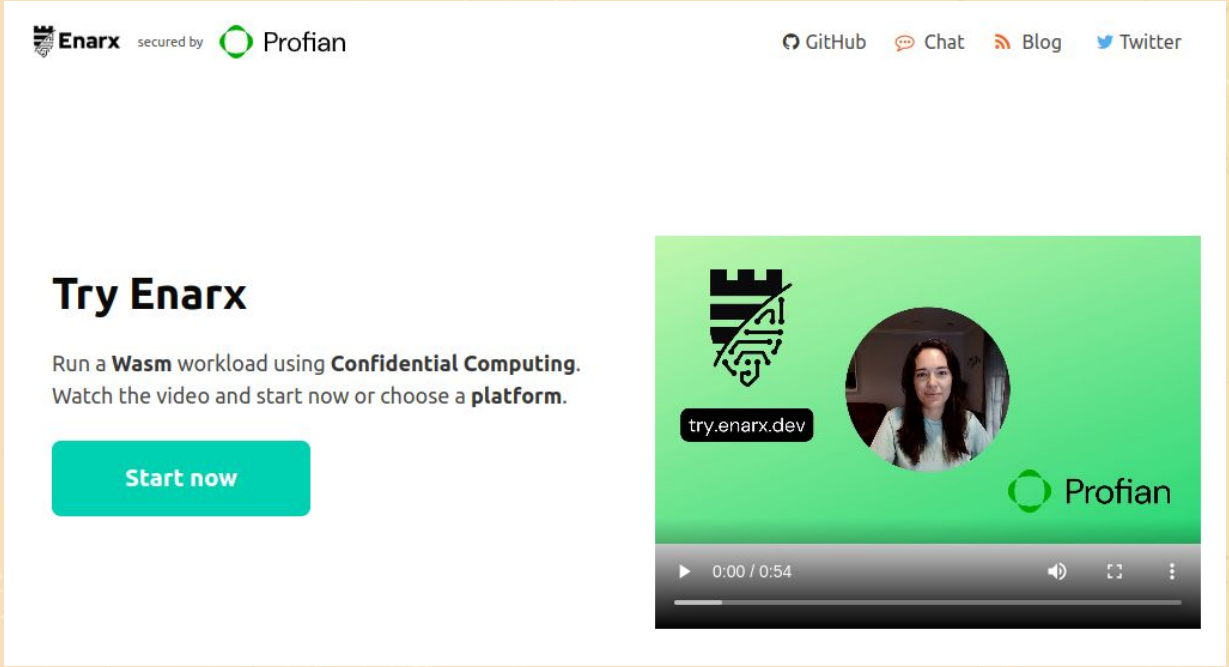
Hosts app & data, but is untrusted

Security Principles of Enarx

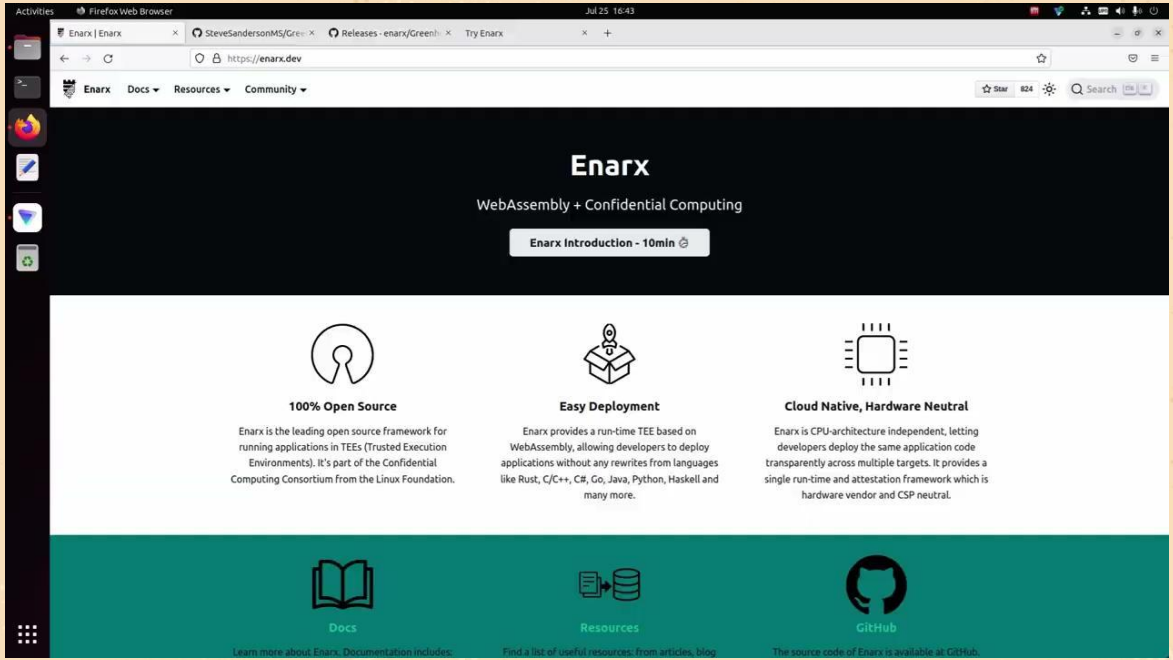
1. Minimal Trusted Computing Base
2. Minimum trust relationships
3. Deployment-time portability
4. Network stack outside TCB
5. Security at rest, in transit, and in use
6. Auditability
7. Open source
8. Open standards
9. Memory safety
10. No backdoors

Demo

Try Enarx @ try.enarx.dev



Greenhouse Monitor



Cryptle

Cryptle

Hack Challenge

THANKS!

Please star our project:
github.com/enarx/enarx

Why WebAssembly?

1. Minimal Trusted Computing Base
2. Minimum trust relationships
3. Deployment-time portability
4. Network stack outside TCB
5. Security at rest, in transit and in use
6. Auditability
7. Open source
8. Open standards
9. Memory safety
10. No backdoors

Why Rust?

1. Minimal Trusted Computing Base
2. Minimum trust relationships
3. Deployment-time portability
4. Network stack outside TCB
5. Security at rest, in transit and in use
6. Auditability
7. Open source
8. Open standards
9. Memory safety
10. No backdoors

Why Open Source?

1. Minimal Trusted Computing Base
2. Minimum trust relationships
3. Deployment-time portability
4. Network stack outside TCB
5. Security at rest, in transit and in use
6. Auditability
7. Open source
8. Open standards
9. Memory safety
10. No backdoors