Enarx & Steward Attestation

Richard Zak // Enarx

28 March 2024

What is Enarx?

Enarx:

- runs WebAssembly (WASI) applications in Trusted Execution Environments (TEEs) under Linux
  - AMD SEV-SNP, Intel SGX/EDMM
  - Others can be supported, like ARM CCA, etc.
- is written in Rust
- can run on Windows, macOS, and ARM Linux without a TEE[1]
- open source project, Apache 2.0 licensed

[1] Without protection, for testing & development

WebAssembly

The use of WebAssembly System Interface (WASI) can be thought of as Posix for WebAssembly, and it provides some benefits for Enarx:

- Portability: The same binary works on different platforms (SGX or SNP, ARM or x86_64, for example). The binary doesn't need to know, or care, which platform is in use.
- Flexibility: Almost any programming language may be used, including C/C++, Go, Rust, Typescript, Ruby, Zig, others.
- Security: Wasi cannot open sockets or files, this has to be done on behalf of the WebAssembly runtime (Wasmtime), so there's no way the program could perform unknown network activity, or "phone home" without the operator's knowledge.
- Confidential Computing provides data security, and the use of WebAssembly transforms the application into data, providing application security.

What is Steward?

Steward is a Confidential Computing-aware Certificate Authority.

- Written in Rust
- Apache 2.0 licensed
- Was a product of Profian, moved to the Enarx org on GitHub, donated to the Linux Foundation
- Can be compiled as a native binary, or as a Wasi application for Enarx
- Steward is stateless, only has the vendors' root CA certificates.
- Steward can generate a random key pair (for testing), or use provided public & private key.
- When workloads are deployed from Drawbridge, Steward adds workload hash to the Certificate (SAN field) as the Drawbridge URL[2].

[2]Was planned, might not be implemented.

What is Drawbridge?

Drawbridge is a Confidential Computing-aware workload repository

- Written in Rust
- Apache 2.0 licensed
- Was a product of Profian, moved to the Enarx org on GitHub, donated to the Linux Foundation
- Can be compiled as a native binary; in the future as a Wasi application for Enarx
- Only releases a workload to Enarx if it authenticates with Steward-signed certificate authentication.
- Uses OpenID to handle user accounts through a provider.

Steward's Attestation Process

Receive via HTTP Post a CSR with an Extension:

- Attestation report
- Vendor CRL
- AMD: CPU cert (Intel has the CPU cert in the Report)
- Intel: TCB report (firmware details)

The following items are checked:

- Does the attestation report signature match the CPU public key and report body?
- Is the CPU's certificate in the PKI chain?
  - Vendor CA $\rightarrow$ intermediate cert $\rightarrow$ CPU cert
- Is the CRL signed by the vendor CA?
- Is the CPU's certificate not in the vendor CRL?
- Optional/configurable items:
  - Minimum CPU firmware?
  - Expected Enarx hash and/or signing key for the Enarx binary?
  - Ensure Enarx hash and/or signing key isn't blacklisted.

HTTP response: Signed cert or error

AMD

The AMD CSR has a few additional items to check:

- Ensure unused parts of the report are zeroed
- AMD has Policy Flags to check:
  - Migration Assistant has to be disabled, we don't want Enarx losing control of it's guest.
  - Debug is rejected if the Steward isn't compiled for debug mode, ensuring a release build won't allow a debug-enabled workload.
- Hashes are SHA-384

Intel

Intel's format is rather different:

- The Intel TCB[3] report is sent as part of the CSR, and requires the "fmspc"[4], which identifies the hardware, and checks:
  - Whether the firmware is updated, or Intel advisories for the firmware.
  - Signing certificate & signature of the TCB.
  - TCB signing certificate is part of Intel's PKI chain.
  - Next update date for the TCB hasn't passed.
  - The signature is valid.
- Steward checks that the FMSPC from the public key matches the TCB report.
- Hashes are SHA-256

[3] Trusted Computing Base

[4] Family-Model-Stepping-Platform Type-CustomSKU

Example Config

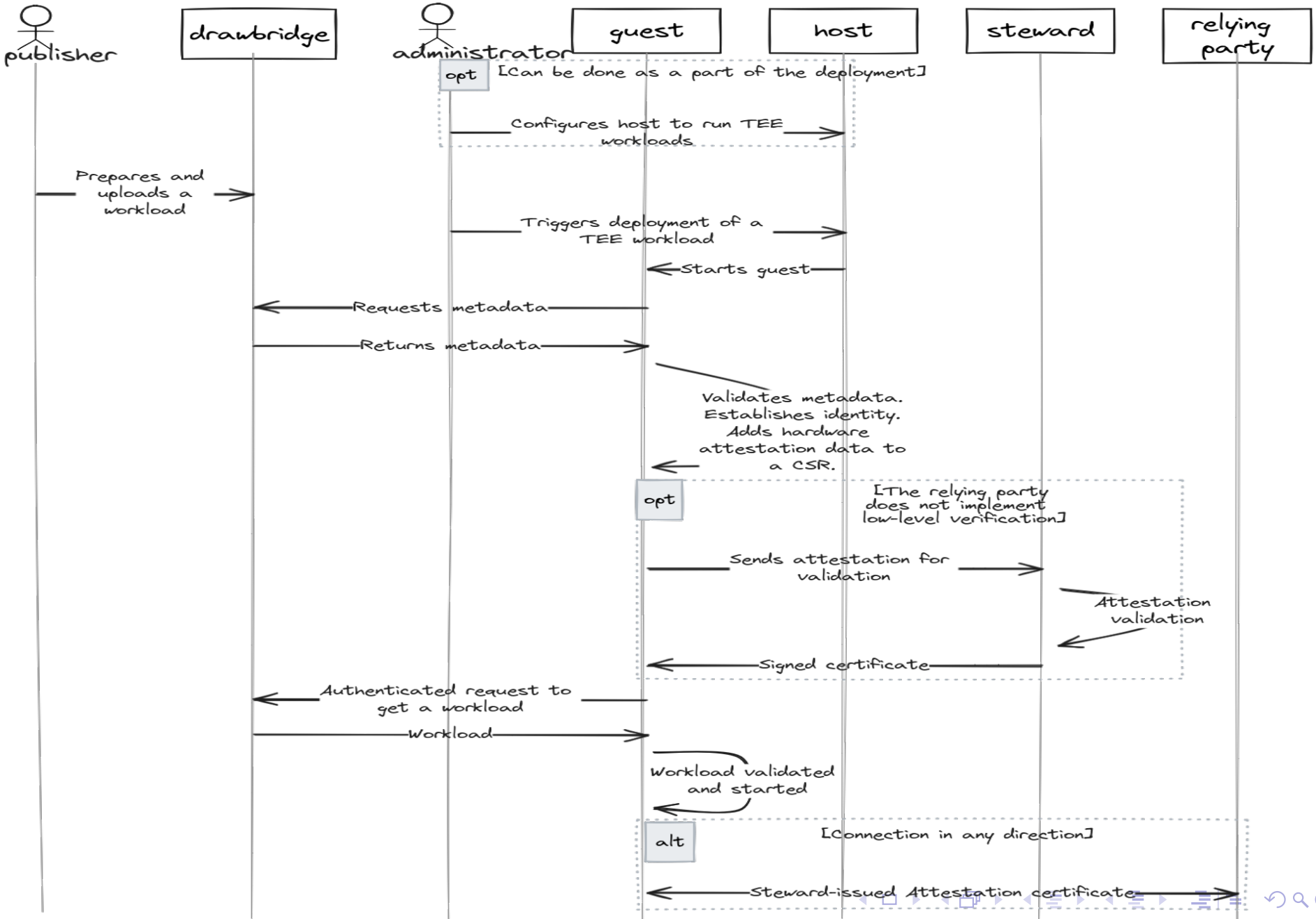
```
[sgx]
signer = ["7a49a07df0f8e90a6e1d9a63e3c696d9c844f0e3f8739b21daa640f99facc48a"]
hash = ["4046ea255f0455131a024e1265ac1ebd131fdbd1240f1712a00381f6f04e8c15"]
features = ["Debug", "ProvisioningKey", "EInitKey", "KSS"]
enclave_security_version = 0
enclave_product_id = 0
allowed_advisories = ["INTEL-SA-00289", "INTEL-SA-00381", "INTEL-SA-00389",
    "INTEL-SA-00477", "INTEL-SA-00586", "INTEL-SA-00614", "INTEL-SA-00615",
    "INTEL-SA-00617", "INTEL-SA-00657"]

[snp]
signer = ["7dc22240a8344fce6ba5f22ffbedabc52d4123ae0ba1c59796e521b953916b503f223b15
id_key_digest = ["d71a4d1da440d515cd69fbb1314acf4221726e82768a8bf6e4a4063ed542ac783
hash = ["ff717ae719840c93c1fca3b7db96488454c3c21b43531488eecff51cfed3febcd91da8be87
abi = ">=1.51" # firmware version, SemVer
policy_flags = ["SMT"]
platform_info_flags = "SME"
```

Workflow

1. Administrator deploys an application from Drawbridge by name & hash
2. Enarx talks to Drawbridge and gets the URL of the Steward
3. Enarx gets the hash of the intended workload
4. Enarx creates an empty Keep
5. Enarx asks the CPU for an attestation report with workload hash
6. Enarx generates a private key, CSR, and adds the attestation report to the CSR as an extension
7. Enarx sends the CSR to Steward
8. Steward validates the CSR and responds with a signed cert
9. Enarx authenticates to the Drawbridge and fetches workload
10. Drawbridge ensures the Enarx cert is in the Steward's PKI chain
11. Drawbridge sends the workload back to Enarx
12. Enarx ensures the workload's hash matches the expected hash
13. Application runs and has Steward-signed certificate for communicating with third parties

Workflow


publisher
drawbridge
administrator
guest
host
steward
relying party
opt
[Can be done as a part of the deployment]
Configures host to run TEE workloads
Prepares and uploads a workload
Triggers deployment of a TEE workload
Starts guest
Requests metadata
Returns metadata
Validates metadata. Establishes identity. Adds hardware attestation data to a CSR.
opt
[The relying party does not implement low-level verification]
Sends attestation for validation
Attestation validation
Signed certificate
Authenticated request to get a workload
Workload
Workload validated and started
alt
[Connection in any direction]
Steward-issued Attestation certificate

Merits of Attestation via CSR

- CSRs & Certs don't require special software on the client end, second/third parties can use existing software.
- No private information about the hardware is leaked.

Drawbacks of Attestation via CSR

The Steward CA has to be trusted:

- added the Steward CA to the operating system's list of CAs, or modify the 2nd party application to only allow this specific CA;
- any configuration of the Steward isn't known to the relying party:
  - any allowed vulnerabilities in the firmware?
  - allowed versions of Enarx?

Acknowledgements

Many thanks to:

- Nathaniel McCallum & Mike Bursell for going out on a limb and creating Enarx, creating Profian, and hiring me;
- Harald Hoyer & Roman Volosatovs for their patient mentorship;
- Nick Vidal for continuing to help Enarx as the Community Manager;
- EuroProofNet for having me;
- AMD and Intel for their exceptional technologies and fantastic documentation; and
- the Confidential Computing Consortium for supporting this technology and sending me.

The Future of Enarx

Enarx development has slowed since Profian closed, but it's alive and there are a few things on the roadmap:

- Recreate `try.enarx.dev`, where people can run their application in a hosted TEE for a limited time using Profian's "Benefice" project.
- Run a public Steward for people to test Enarx with an Enarx CA.
  - Possibly run a demo or debug Steward with looser restrictions.
  - URL: `ca.enarx.dev`
  - HSM integration (need to buy & learn how to use an HSM)
- Run a public Drawbridge.
- Continue work on the VFS project for Enarx:
  - Network connection policies
  - Read-only on the host unencrypted
  - Transparently encrypt writes to local disk.
  - Virtual file operating for managing crypto operations.
  - Virtual file operations for opening new network connections
- Continue to promote Enarx online and in-person.
- Keep Enarx updated with changes in WebAssembly.

The Future of Enarx

Accomplished post-Profian:

- ✓ Received & configured CI servers so Enarx commits are again tested with SGX & SNP.
- ✓ Merged in support for AMD SNP v10 patches (thanks Tom Dohrmann!)
- ✓ Steward & Drawbridge relicensed, moved to Enarx org.
- ✓ Keeping the projects' dependencies updated (in progress).
- ✓ Keeping Enarx's dependency crates[5] created by Enarx/Profian updated and providing releases (on-going).
- ✓ Updates and content for the website (on-going).
- ✓ Gain control of social media accounts (Mastodon, LinkedIn) for Enarx.

[5] Ciborium, Crt0stack, Flagset, Sgx, Vdso, etc

Attestation Workflow Mermaid