Reproducible builds for confidential computing: Why remote attestation is worthless without it
Confidential Computing

- We trust no one!
- Well, beside hardware manufacturer
- Need to establishing trust in the rest
- Remote attestation
RFC 9334: Remote ATtestation procedureS (RATS)
Reference values in Confidential Computing
What’s part of my Trusted Computing Base?

- CPU
  - Microcode
  - Firmware

- CVM
  - SVSM / HCL
  - Firmware
  - Bootloader
  - Kernel
  - Userspace
Who’s part of my Trusted Computing Base?

- CPU
  - Microcode
  - Firmware

- CVM
  - SVSM / HCL
  - Firmware
  - Bootloader
  - Kernel
  - Userspace
Remote attestation without reproducible builds

- Every trusted software vendor can run an attack
  - Delivers reference value of malicious binary
  - We can only check the authenticity
  - No insight what code is running
- What if we build everything from source?
  - We are the remaining software vendor that needs to be trusted
- Actual goal: attestation through the *end-user*
Reproducible Builds

“Reproducible builds are a set of software development practices that create an independently-verifiable path from source to binary code.”
Reference values in Trusted Computing - Expectation

\[ \text{referenceValues} = f(\text{sourceCode}) \]
Reference values in Trusted Computing - Expectation

- Source Code
  - Firmware
  - Kernel
  - Userspace
  -...

Reference Values
Reference values in Trusted Computing - Reality

- Closed Source
  - Firmware
  - Kernel
  - Userspace
  - ...

Reference Values
Reference values in Trusted Computing - Reality

Timestamp → Source Code → Randomness → Undeclared inputs → Compiler toolchain

→ Firmware → Kernel → Userspace → ...

→ Reference Values
Reference values in Trusted Computing - Reality

Evidence of boot x ≠ Evidence of boot y
Positive examples

- github.com/aws/uefi
- github.com/edgelesssys/constellation
- github.com/confidential-containers/cloud-api-adaptor (podvm-mkosi)
- github.com/edgelesssys/reproducible-mkosi
Reproducible mkosi

Build bit-by-bit reproducible OS images
How to build reproducible OS images

- Nix provides hermetic build tools
- Pin distro packages with lockfile
- Build in a Sandbox (mkosi, Nix(OS), Bazel)
- Restrict build actions (do not use Hashicorp Packer or Dockerfile)
Thanks!

- Learn about reproducible builds: reproducible-builds.org
- Provide an open software stack for CC
- Enable the community to reproduce reference values

Malte Poll
@malte@chaos.social
@malt3
github.com/malt3

Paul Meyer
@katexochen@infosec.exchange
@katexochen
github.com/katexochen