Securing the IoT Supply Chain with DevSecOps

Daniel Morrison & Antonio Escalera on behalf of Raft
Objective

- Provide a look at some of the most impactful DevSecOps approaches/technologies available and explore how IoT manufacturers may strategically implement them to augment their security posture.
- NOT to provide a complete, production-ready solution.
Develop with Security in Mind

- Security is built from the first commit, so sign them.
- Adhere to the *principle of least privilege* within your source code repositories; configure your branches and organization to protect them from bad actors.
- Establish an agreed upon *definition of secure* and *threat tolerance* for ingesting open-source dependencies; keep them up to date and secure.
- Consistently perform SAST to ensure accepted levels of threat are not breached.
Securing the Build Process

- *Where* and *how* you build matters (a lot), make the right choice for your Supply Chain Levels for Software Artifacts (SLSA) needs
- Generate provenance for every build which produces artifacts and make that provenance available alongside those artifacts
- All OCI images produced should be signed and be made available alongside their (attested) SBOM
03 Update Automation &amp; Pipelines

- GitOps for applications
- GitOps for infrastructure
- GitOps for firmware
- Sporadic Connectivity versus Always-On
04  **Network Layer: WWAN & WLAN**

- OTA updates, monitoring, control
- Security throughout layers
- 5G-AKA/EAP-TLS, IPSec, 802.1X
- Encryption & access-control – confidentiality & integrity
- Network segmentation/application brokers
- Traffic monitoring – NetFlow
- Continuous evaluation – OpenVAS, OpenSCAP
Remote Attestation & Field Registration

TPM-enabled Remote Attestation Protocol (TRAP)

Devices ship with:
- Device-specific enrollment key
- Bootloader hash
- Printed registration code

Customer boots device:
- TPM verifies bootloader contents
- Bootloader hashes application content/rootfs
- User connects to device and inputs registration code
06 Secure Boot

- Root of trust in secure element
- Bootloader verifies authenticity and integrity
- Signing and verification of firmware and software
- Chain of trust between components
- Updates and patching
Trusted Execution Environment

- Separate, isolated environment for code execution
- Guaranteed confidentiality/integrity for code/data
- Hardware-based isolation
- ARM TrustZone – Cortex-A and Cortex-M
- Secure boot chain & root of trust
- Communication between worlds – Secure Monitor & SMC
Trusted Platform Modules

- Storage area, crypto processor, security functions
- Tamper-evident and resistant
- Integrated into system board, usually separate chip
- Secure storage of root of trust
- Firmware and bootloader integrity verification
- Measure system configuration and software
- Enforce device-level security policies and monitor health
Service Meshes & mTLS

- Provides additional assurance that inter-service network traffic is being authorized from both the client and server.
- This is especially important for many IoT devices which do not follow ironclad login procedures.
- Utilize a service mesh to enforce strict adherence to mTLS across your application suite, to diminish and outright eliminate your risk of various malicious attacks.
Admission Controllers

- Cosign Policy Admission Controller can protect namespaces in your Kubernetes cluster by ensuring any scheduled OCI image was signed using a known key.
- The policy controller can also perform policy-as-code verifications on attested payloads using Rego or Cue.
- Additional admission controllers such as Gatekeeper or Kyverno can be used to enforce that workloads are only schedulable within policy controller protected namespaces.
Demo

Signed and attested with CVE-free SBOM
Any Questions?
Together, we’ll make waves

www.goraft.tech