MontaVista

Solution Brief

Securing embedded designs with “CIA”

Confidentiality, **I**ntegrity & **A**ccess-control : Cybersecurity@Edge

Smart, inter-connected devices are chaining worldwide networks into configurable services. However, this new avenue is prone to security and trust challenges, leaving the chain as good as its weakest link.

To make an embedded product safe from malicious attacks the hardware and software present in the device must work together to enable robust security countermeasures. This solution briefs tries to

- Propose Challenges and Approach for delivering proactive security to withstand known and zero day attacks.
- Define Architecture, Security, and Device Maintenance for an ARM 64 based secure gateway reference design (Cybersecurity @edge))using Cavium™ Octeon Tx (81xx) SoC and MontaVista’s Carrier Grade eXpress (CGX) Linux.

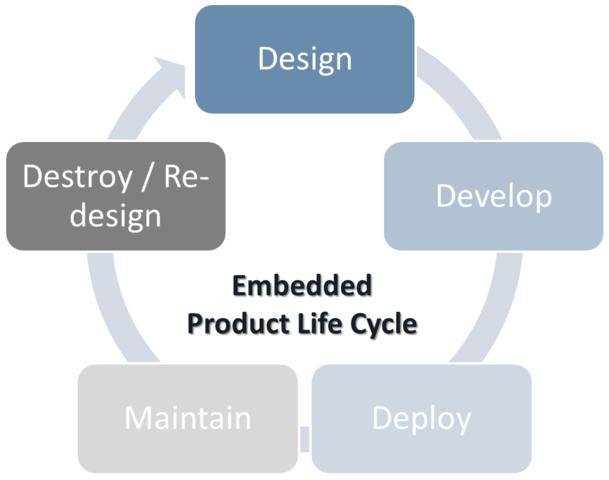


Fig 1: Secure embedded PLC

MontaVista Carrier Grade eXpress provides necessary software, tools and support to help custom designs, by:

- **Confidentiality :** Enabling “Root of Trust”, with “Secure Boot” & “Secure Update” using Hardware (TrustZone TEE, TPM) for encryption key management. Network Security features (SSH, IPSEC, Firewalls & DPI including platform specific Hardware Off-loads) and/or “Block Level” encryption using dm_crypt rootFS.
- **Integrity:** Integrity here means not just unchanged, but "unchangeable", or "immutable" and it requires a system wide “Root of Trust” to ensure this. Linux Kernel security subsystem provides for an Integrity Measurement Architecture (IMA), which focuses on the validation of file integrity before these files are loaded (and perhaps executed). Alongside IMA is the Extended Verification Module (EVM) subsystem, which provides protection against tampering the hashes themselves
- **Access-Control:** Linux Kernel Security (SELinux vs AppArmor vs Grsecurity) provide for a mechanism for supporting access control security policies, including Mandatory Access Control (MAC). In addition, MontaVista CGX incorporates continuous Vulnerability (CVEs) tracking and updating to ensure a hardened Linux distribution that is regularly maintained.

BENEFITS

- OUT-OF-THE-BOX EXPERIENCE WITH PRE-TESTED BSP
- CARRIER GRADE RELIABILITY AND HIGH AVAILABILITY
- VULNERABILITY TRACKING & SECURE KERNEL AND APPLICATION UPDATE
- Smart Edge with KVM and readymade VNFs

APPLICATIONS / USE CASES

POSSIBLE USE CASES

- 5G CARRIER GRADE INFRASTRUCTURE
- IOT AND SMART GATEWAY
- SMART HOME AMONG OTHERS

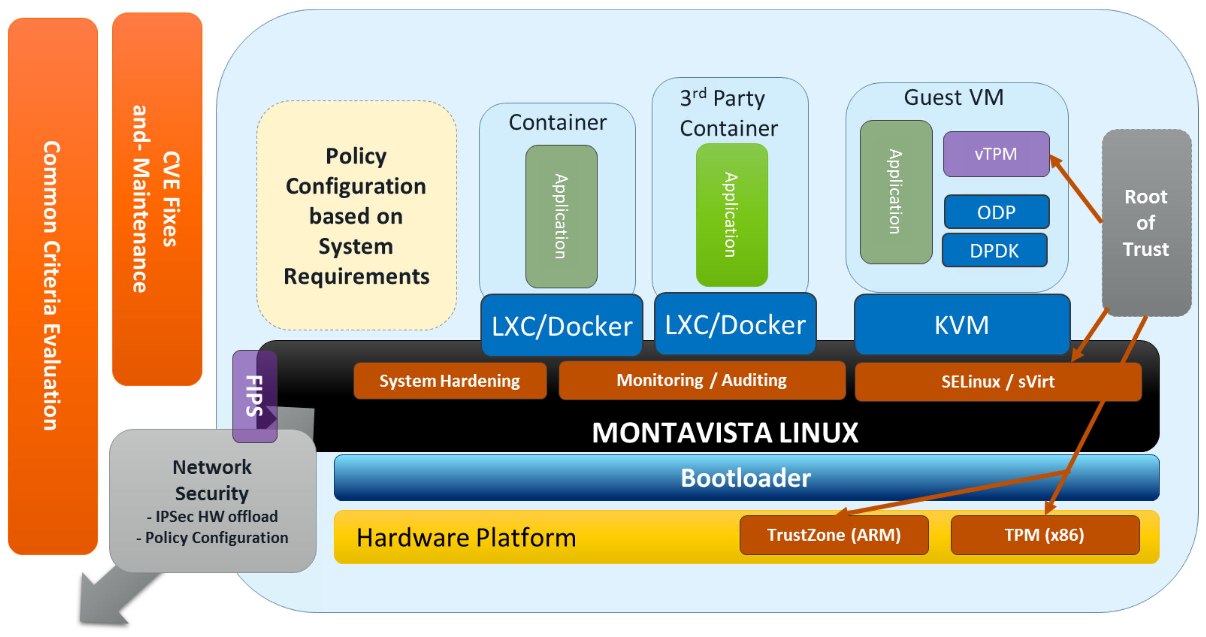


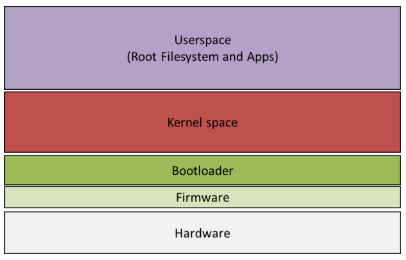
Fig 2: Architectural Design Consideration with MontaVista CGX®

Cybersecurity threat and Secure product

Cybersecurity challenges for edge connected devices are on the rise, some experts say exponentially. Linux and ARM offer native technologies readily available to help create solutions that secure onboarding and enable cryptographic security. Developers today then have the means to architect and deploy secure edge devices.

Security from Bottom up

A typical boot cycle of a device involves the following steps including its associated software.



- As soon as the device is powered on, the firmware comes up and loads the bootloader
- Then the bootloader boots the kernel
- Kernel then loads the necessary modules and brings up the users-pace programs or applications.

From the above it is quite clear that we need to have a multi-layered security approach with various tools as each level.

Secure boot

Secure boot ensures only authenticated software runs on the device and is achieved by verifying digital signatures of the software prior to executing that code. To achieve secure boot, hardware based mechanism i.e. those enabled in processor/SoC are required.

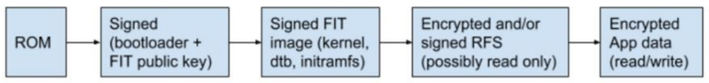

ROM
Signed (bootloader + FIT public key)
Signed FIT image (kernel, dtb, initramfs)
Encrypted and/or signed RFS (possibly read only)
Encrypted App data (read/write)

Fig 3: Chain of Trust

A security friendly Cavium® Octeon TX™ provides Authentik™, a technology that allows the whole multi-core processing chip to be locked; so that OEMs that entrust third parties to assemble their systems can be assured that no rogue or counterfeit copies of their systems .

Chain of trust

Extending the trust scheme all the way to user space involves establishing a chain of trust i.e. ROM verifies signed bootloader, bootloader verifies signed kernel and it verifies encrypted/signed root filesystem (RFS).

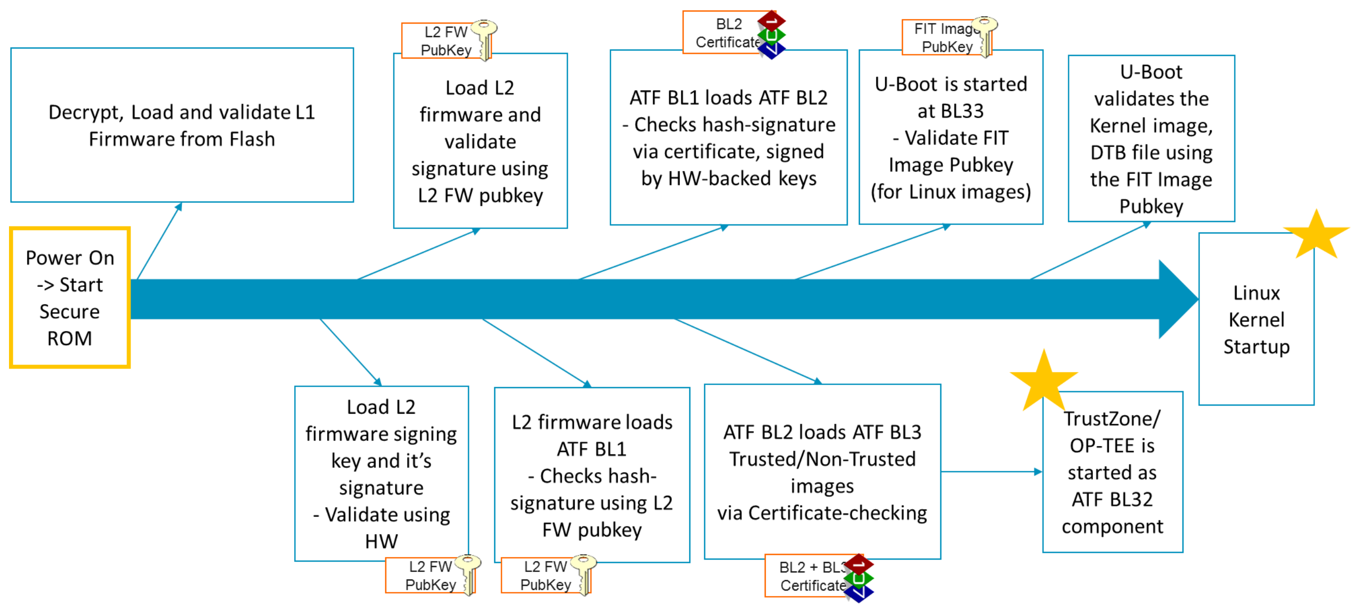

Power On -> Start Secure ROM
Decrypt, Load and validate L1 Firmware from Flash
L2 FW PubKey
Load L2 firmware and validate signature using L2 FW pubkey
BL2 Certificate
ATF BL1 loads ATF BL2 - Checks hash-signature via certificate, signed by HW-backed keys
FIT Image PubKey
U-Boot is started at BL33 - Validate FIT Image Pubkey (for Linux images)
U-Boot validates the Kernel image, DTB file using the FIT Image Pubkey
Linux Kernel Startup
Load L2 firmware signing key and it's signature - Validate using HW
L2 FW PubKey
L2 firmware loads ATF BL1 - Checks hash-signature using L2 FW pubkey
L2 FW PubKey
ATF BL2 loads ATF BL3 Trusted/Non-Trusted images via Certificate-checking
BL2 + BL3 Certificate
TrustZone/ OP-TEE is started as ATF BL32 component

Fig 4: Summary, Secure Boot and Key Management on the reference HW

Let's explore some of the methods to secure each of the above.

Bootloader authentication

Bootloader authentication is processor specific. However the high level mechanism is usually the same, it involves:

Creating a public/private key pair

- Signing the bootloader using vendor-specific code signing tools
- Applying the vendor-specific configuration on the HW to enable initial checking of the firmware image

The processor ROM code on power-up loads the bootloader along with the signature/certificate appended to it. It then verifies the software by performing the following steps:

- Verify the public key used in the signature/certificate with the one applied at the HW setup phase
- Extract the hash of bootloader from the signature using the verified public key
- Compare the extracted hash with the computed hash of the bootloader. If it matches it proceeds with the boot process, thus authenticating the bootloader.

FIT image authentication

FIT stands for "Flattened Image Tree" and is a single binary that can contain multiple images along with metadata for storing image information along with signatures, thus making it convenient to authenticate.

- A typical secure boot use case is to generate a FIT image containing kernel, device tree and initramfs.
- The FIT image is then signed using a private key, and the signature is embedded inside the FIT image.
- The public key is then embedded inside U-Boot as part of U-Boot device tree.

Since the signed U-Boot is authenticated by the ROM, we can trust the public key inside of U-Boot to verify the FIT image.

Block Level or Full Disk Authentication using dm_crypt

As an alternative to FIT image, single boot image can be created, and encrypted with SoC vendor specific APIs.

- Only if a root filesystem (RFS) is read only and small enough to run out of RAM, then embedding the root filesystem inside the FIT image should be sufficient for authenticating it.
- However, typical RFS are large, and we need the mechanisms provided by the Linux kernel for authenticating its contents. This is performed by the kernel's device mapper (dm) modules.

Linux Integrity Management

“Integrity” means not just unchanged, but "unchangeable", or "immutable" and it requires a system wide “Root of Trust” to ensure this.

The **Integrity Measurement Architecture** (IMA) is mainly a Linux kernel driven security subsystem which focuses on the validation of file integrity before these files are loaded (and perhaps executed). It does so by comparing the hash of the file with a stored hash. The technology is specifically designed for preventing offline tampering of data i.e. either the hash is cryptographically secure, ensuring that no-one can generate the same hash without access to a private key, or the validation is done remotely (called remote attestation) where the remote side checks the measured hashes against its own list. IMA follows Trusted Computing Group (TCG) open integrity standards.

Alongside IMA is the **Extended Verification Module** (EVM) subsystem, which provides protection against tampering the hashes themselves, as well as tampering other extended attributes of files, such as SELinux extended attributes).

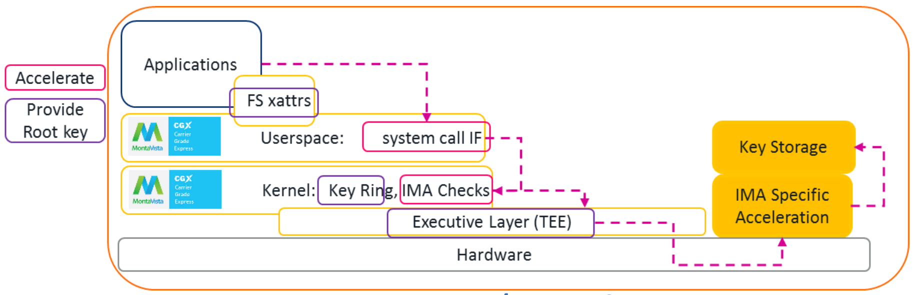


Fig 5: IMA/EVM for Integrity

Together, these technologies play an important role in Linux adoption of the Trusted Computing Base, interacting with specially designed security devices on systems (such as the Trusted Platform Module or TPM chip).

ARM TrustZone and Trusted Execution Environment (TEE)

ARM has included TrustZone-enabled IP in the cores that allows for setting up trusted devices, memory areas, and even trusted interrupts at system boot. It allows running a “secure world” operating system or executive, enabling a degree of virtualization and isolation supported by hardware separation. In case of ARM TrustZone enabled devices, TPM can be replaced by TrustZone which also would result in BOM cost saving.

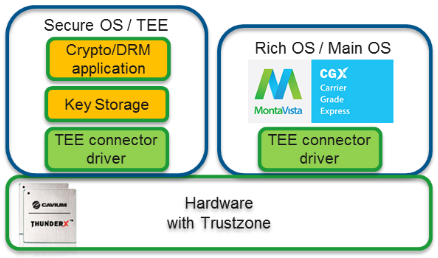


Fig 6: TEE & ARM TrustZone

A Trusted Execution Environment can run inside TrustZone or in a similar environment that can fence off the secure components from the main system. TEE itself is a Global Platform standard for such operating environments. Available implementations include for example SierraTEE and OP-TEE (supported by Linaro).

OP-TEE (Open Platform - TEE) is an open source implementation of TEE. This project contains a full implementation to make up a complete Trusted Execution Environment.

- Optee_OS - This component meets the Global Platform TEE System Architecture specification. It also provides the TEE Internal core API v1.1 for the development of Trusted Applications.
- Optee_client—This component provides the TEE Client API. There are two main target/binaries to build. There is libteec.so, which is the library that contains that API for communication with the Trusted OS. Then the other target is the binary tee-supplicant which is a daemon serving the Trusted OS in secure world with miscellaneous features, such as file system access.
- Optee _test— TEE sanity test suite in Linux using the ARM TrustZone technology. It is distributed under the GPLv2 and BSD 2-clause open-source licenses

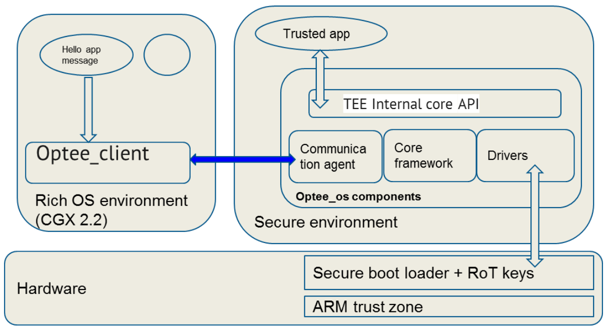


Fig 7: OP-TEE Framework

MontaVista

Solution Brief

Securing embedded designs with “CIA”

Confidentiality, **I**ntegrity & **A**ccess-control : Cybersecurity@Edge

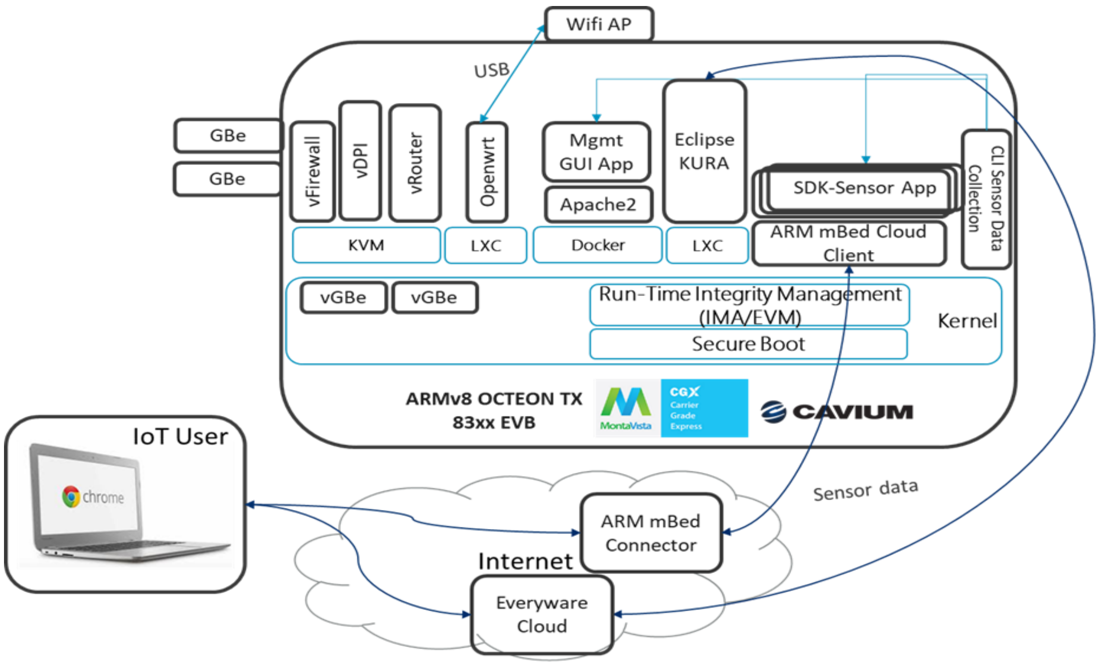


Fig 8: Case Study: Cybersecurity@EDGE

CYBERSECURITY@EDGE: SECURE IoT Gateway Reference

The issue of safety and security is specially relevant in the case of the internet of things, involving consumer applications which can be potentially hacked. Communications between the things, the gateway, and the cloud service must be cryptographically secured to preserve confidentiality, integrity, and authenticity.

Secure Boot, IMA/EVM, “Authentik” & OP-TEE framework

If an attacker were to compromise the IoT gateway, not only the data passing through the gateway is at risk, but control of the physical things connected to it are at risk as well. By implementing a true end-to-end, security solution that ensured a system wide “Root of Trust” and a “Trusted Execution Environment” for applications, we demonstrate the capability to meet the need for security in a modern connected IoT device/gateway.

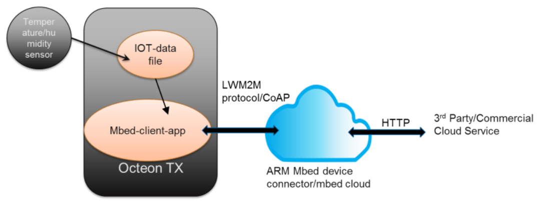
Virtualization and Virtual Network Functions (VNFs)

Another objective for this solution demonstration is to showcase use of open virtualization technologies like KVM, LXC/Docker & Kubernetes (Production-Grade Container Orchestration) for isolating Virtual Network Functions (VNFs), applications and system software

Commercial Device Cloud Integration

ARM mbed IoT Device Platform provides operating system, cloud services, tools and developer ecosystem for creation and deployment of commercial, standards-based IoT solutions. It is made up of components such as device software and cloud based device management services that enable movement of data from sensor to server.

The demonstration has mbed Linux client running on ThunderX gateway that reads data from a cluster of such devices (ex: Temperature and Humidity sensors from outside world) in real time. The data is then encapsulated into a resource using client libraries and published to Mbed cloud or Mbed device connector using LWM2M (Light Weight Machine 2 Machine) protocol. Security key is also embedded so that mbed cloud registers it as valid device through registration services/APIs. The sensor data is now made available on the med-device-connector or on mbed-Cloud.



Temperature/humidity sensor
IOT-data file
Mbed-client-app
Octeon TX
LWM2M protocol/CoAP
HTTP
3rd Party/Commercial Cloud Service
ARM Mbed device connector/mbed cloud

Fig 9: ARM mBed Cloud

MontaVista

Solution Brief

Secure IoT gateway solution based on CAVIUM OCTEON TX™ is a simple, secure and scalable prototype. It supports a wide array of IoT optimized interfaces engineered to support the secure delivery of IoT services to a wide base of customer use cases.

eXpress.Connected.Everything.

About MontaVista Software

MontaVista Software, LLC, a wholly owned subsidiary of Cavium Networks (NASDAQ:CAVM) is a leader in embedded Linux commercialization. For over 15 years, MontaVista has been helping embedded developers get the most out of open source by adding commercial quality, integration, hardware en-ablement, expert support, and the resources of the MontaVista development community.

MontaVista All Ways Open

MontaVista Software
2315 North First St, 4th FL
San Jose, CA 95131
Email: sales@mvista.com
Tel: +1-408-943-7451