Qualcomm Developer Network Presents

Developing for Industrial IoT with Embedded Linux OS on DragonBoard™ 410c by Timesys University

Co-sponsored by Qualcomm Technologies, Inc. and Arrow Electronics

Embedded Linux from a Trusted Source

timesys®

Session 4
Embedded Products Security

Maciej Halasz, Vice President of Technology
Timesys Corporation

Dieter Kiermaier, Senior Field Application Engineer
Arrow Electronics

Webinar Series

- **Session 1:** Introduction to DragonBoard 410c SoC and Starting Development of Your Embedded Linux based “Industrial Internet of Things” (IIoT) Device
  - Setup for designing IIoT products
  - How to assemble and deploy initial BSP
- **Session 2**: Application Development for Embedded Linux
  - Application development environment setup
  - How to reflect product requirements in the BSP
  - Communication in the IIoT system
- **Session 3**: Building a Cutting-Edge User Interface with Qt®
  - Developing modern, rich UIs for factory terminals
- **Session 4**: Embedded Products Security
  - Designing security-rich devices

Session 3 Recap

- **What we did**
  - Discussed the MQTT protocol as an example of inter-device communication
  - Talked about the Qt Software
    - How to integrate it into an OpenEmbedded RPB BSP
    - How to build an SDK with Qt5 libraries and utilities
    - How to setup DragonBoard 410c SDK in the QtCreator
- **Qt Software**
  - Packages: Application, Device Creation, Boot to Qt®
  - What modules are in Qt software
  - How Qt is licensed
  - How to develop and deploy an application on a DragonBoard 410c
  - Qt for Automation
- **Key takeaways**
  - MQTT is a lightweight, energy efficient protocol for IoT
  - Qt Software declarative programing allows for slick animated UIs with gesture controls
  - Qt Software is available within OpenEmbedded RPB BSP for the DragonBoard 410c

Session 4 — Agenda

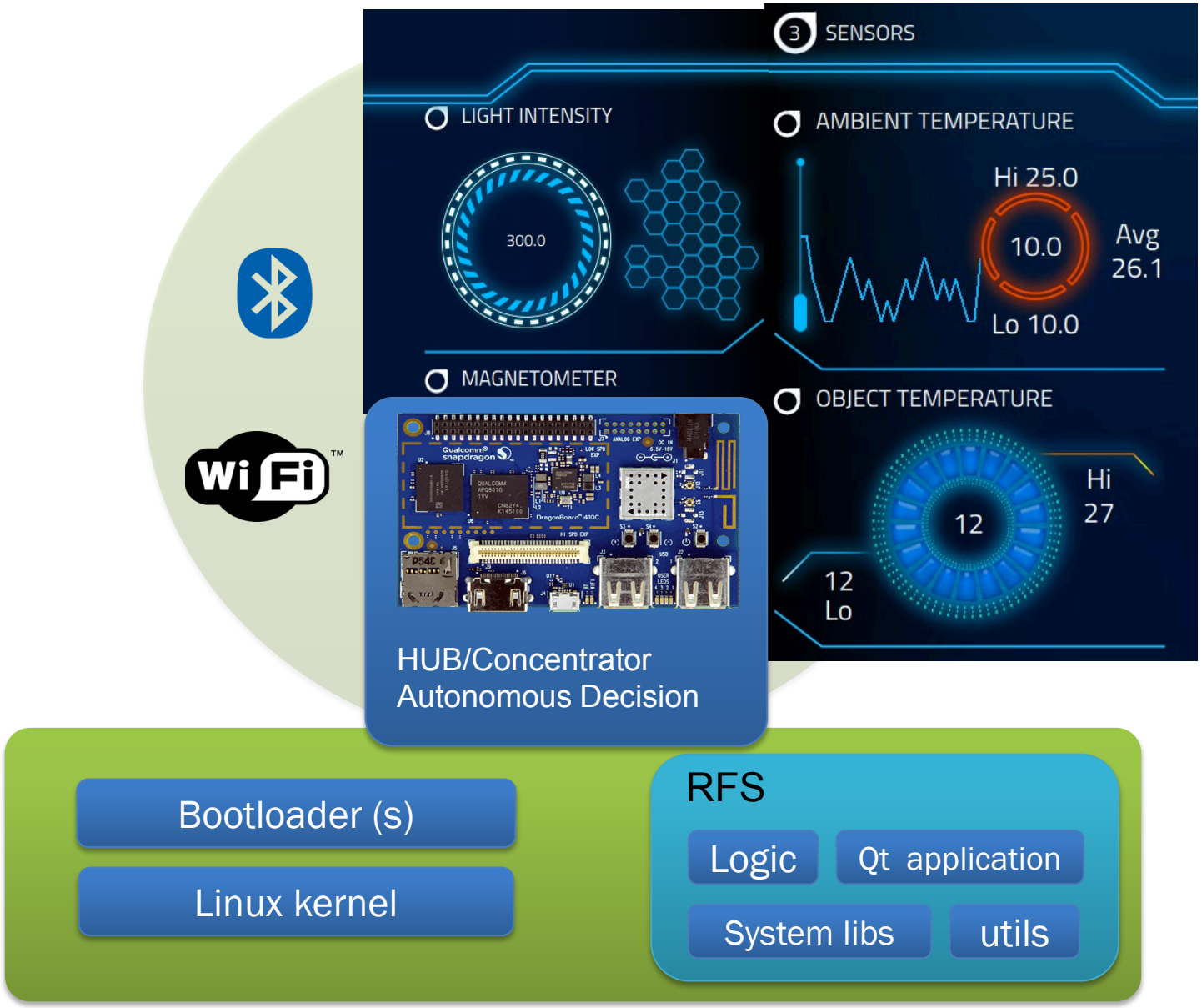
- **Is security important? Why?**
- **Security in an embedded Linux device**
- **Techniques to help secure embedded Linux product**
  - Signing
  - Encryption
- **Security – an ongoing challenge**
  - Timesys security feed
- **Trusted Platform Module**
- **Q & A**

Is Security Important?

Why Is Security Important?

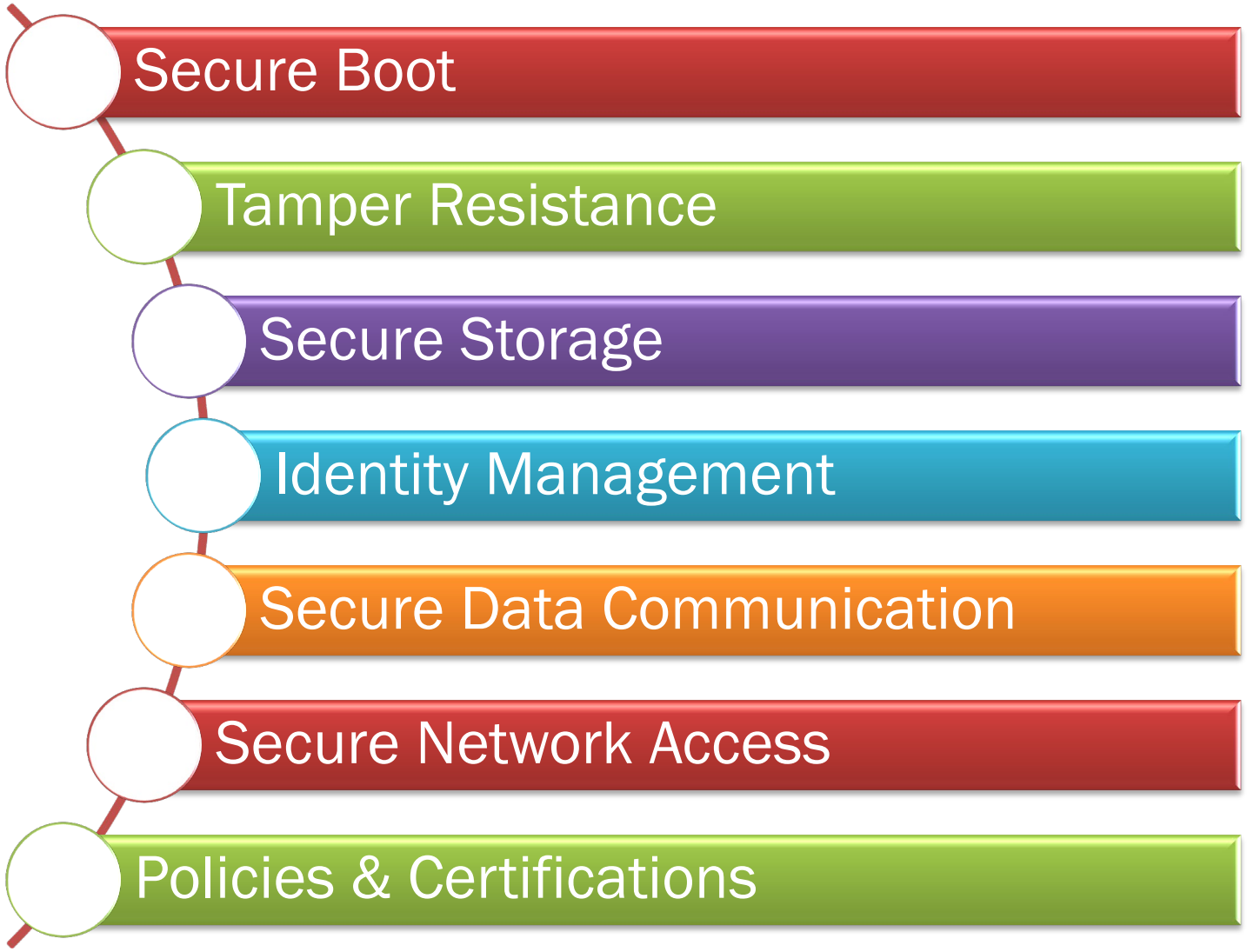
- **Whether a commercial embedded OS or open source, no code is threat proof.**
- **The manual process of securing device is not feasible.**
  - Requires time and effort to maintain an embedded Linux distribution with patches and security vulnerability protection
  - Rate of information-security vulnerabilities is increasing and discoveries are unpredictable
- **… Therefore, you need a process that:**
  - Performs security audit for comprehensive security
  - Performs system hardening
  - Continuously monitors vulnerability discoveries
  - Notifies your engineering development team of the vulnerabilities
  - Assesses discovered vulnerabilities against the software components used in the team's build to verify applicability
  - Applies and tests the fixes, and deploys the updated software in an efficient way

Linux Based Device Structure



- **Different attack vectors:**
  - Direct connection to the system
    - USB, CAN, Expansion ports
    - debug header
  - Access to storage
    - SD Card
    - Hard Drives
  - Access to memory
  - Network access
    - TCP/IP, WIFI
  - Console on UART
  - Bootloader settings access
    - NFS mount enabled

Device Security Layers



Security is an elusive target as it is a function of money and effort.

Upfront security solves the initial challenge; ongoing security keeps the device protected.

Embedded Linux Security Processes

Prototype

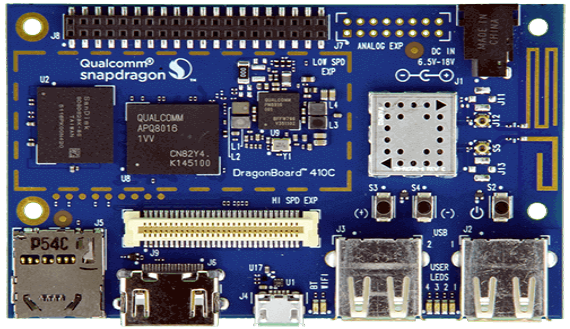
- Threat modeling
- Data/OS encryption
- Authentication
- Secure boot and update
- Encrypted data transfer
- Trusted software
- Mechanical/electrical

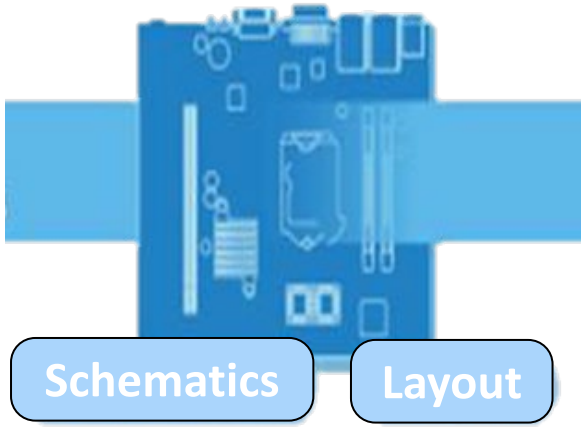
Develop

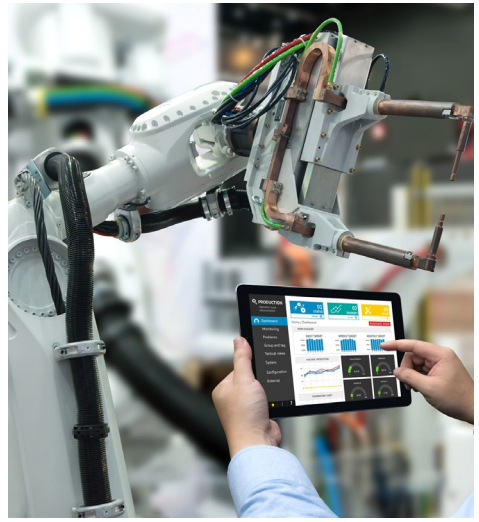
- Board bring-up
- Partitioning
- Secure boot
- Patching, upgrading
- Key management

Production

- Exploit monitoring
- Patching & updates
- Vulnerability scan
- Penetration testing
- Kernel/OS upgrade readiness







Qualcomm Snapdragon is a product of Qualcomm Technologies, Inc.

timesys®

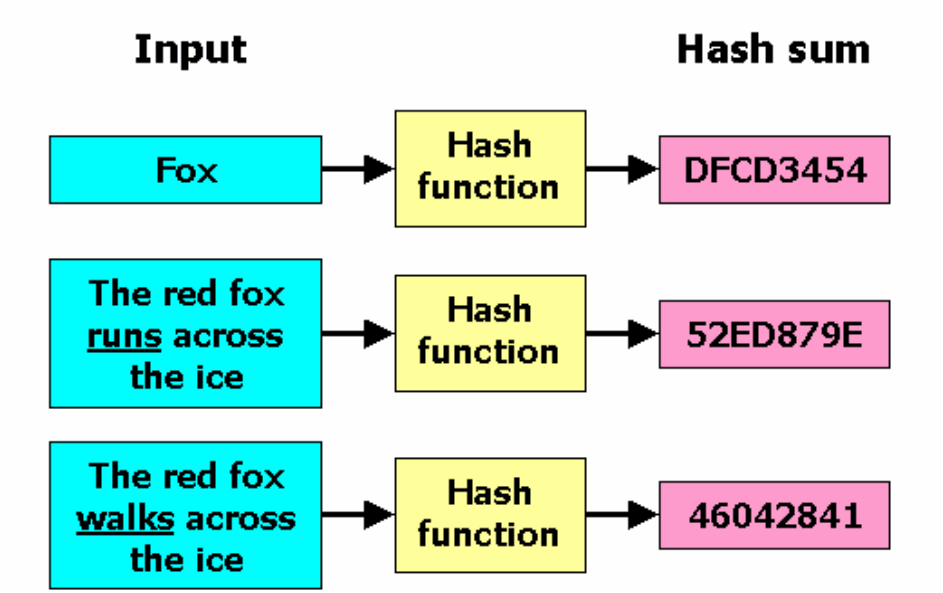
Security Audit — Device Software Centric

- **Physical Assessment**
  - Is the device protected against unauthorized access?
  - Are peripherals protected?
- **Access Control Assessment**
  - Who all (users/programs) have access to the device and with what level of permissions?
  - Is access being logged?
- **Vulnerability Assessment**
  - What software is on the device and what are the known vulnerabilities?
  - How are we notified about security issues?
- **Network Security**
  - Is the data being sent to the authorized server? How to verify server identity?
  - Is the data transmitted securely?
  - Are the web server access and other network services secured?
- **Software Update Process**
  - How do we fix and securely deploy updates?
  - How do we know the device is running our firmware?
  - Is the software update being received from a trusted source?
  - Is software update integrity not compromised?

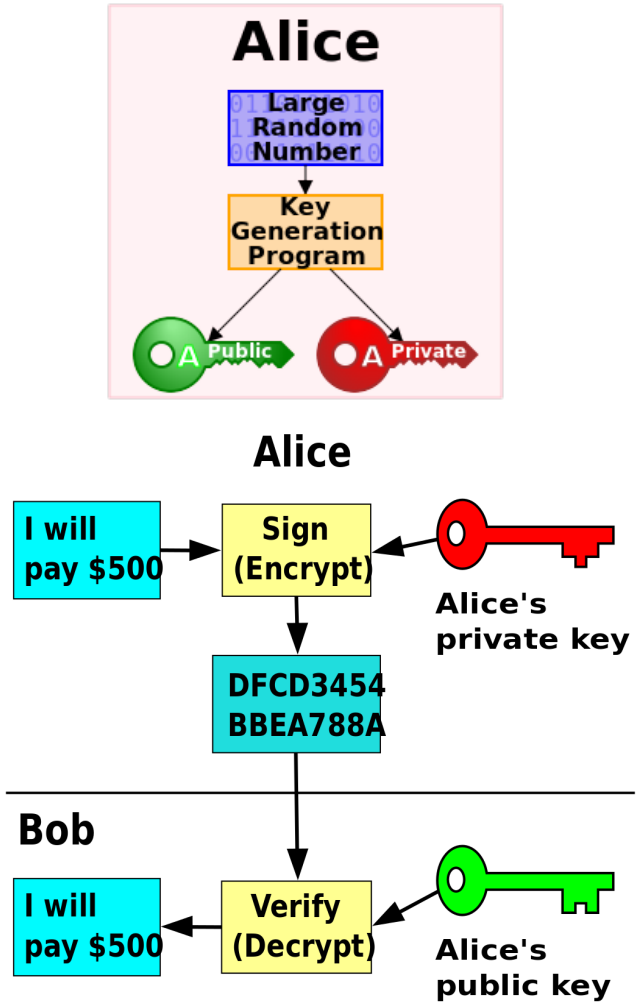
Signing Images — Authenticating Software

Terminology

- Hash



- Public key cryptography

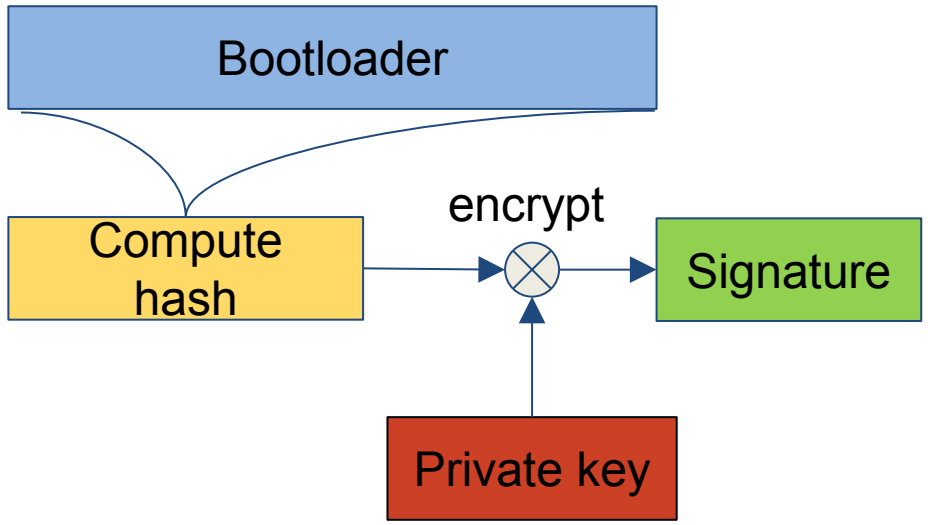


Secure Boot Without Encryption

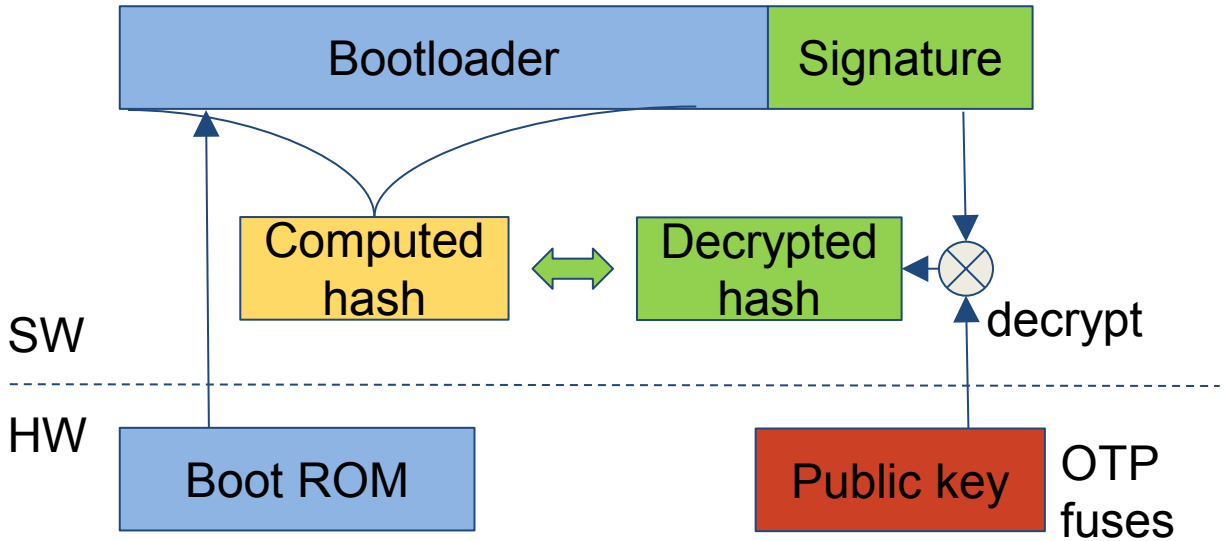
- **Provides**
  - Authentication (unauthorized images not allowed to run)
  - Integrity (authorized images can not be ‘tampered’ with)
- **Does not provide**
  - Anti-cloning
- **Uses asymmetric key for signing**
  - Private key -> used for signing
  - Public key -> used to verify signature
- **Bootloader verification performed by ROM code**
  - SoC specific

Secure Boot Flow

Host PC



Device



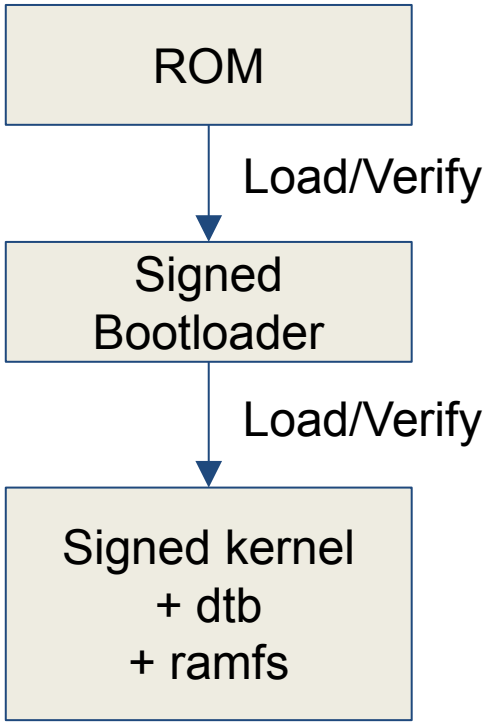
Hash must match to boot!

Chain of Trust

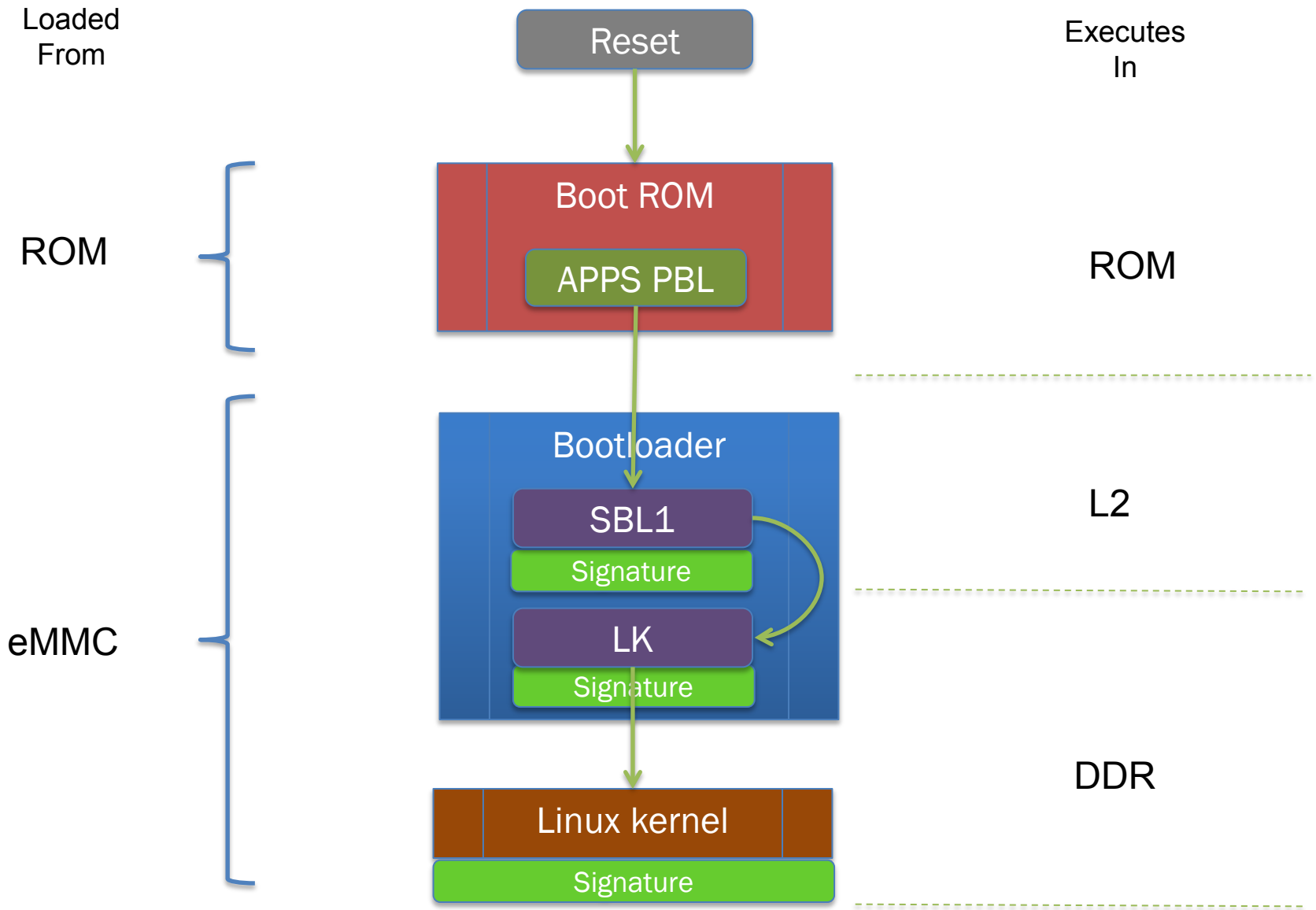
- **The whole software needs to be authenticated and validated — not just the bootloader**
  - Single failure along the chain will render the process insecure
- **Extending secure boot scheme to user space**
  - ROM
  - Bootloader (eg: SBL and/or Little-Kernel)
  - Kernel/Device tree
  - RFS

SoC features (kernel, dtb and/or ramfs)

- **Verified bootloader uses ROM API's to check kernel/dtb/ramfs signatures**
- **Drawbacks**
  - SoC specific code might not be mainlined
  - Needs vendor specific signing tools, format
  - Not integrated with build systems
  - Limited to RAMFS (read only / size limited by RAM)



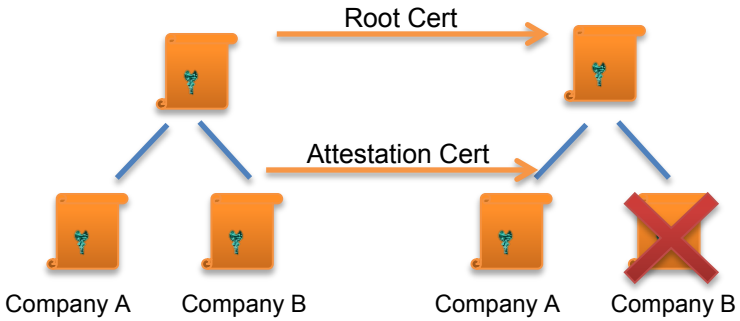
Secure Boot Process on the DragonBoard 410c



Signing Tools

SecImage Tools are available to customers designing with Snapdragon 410E – on request through Arrow Electronics representative

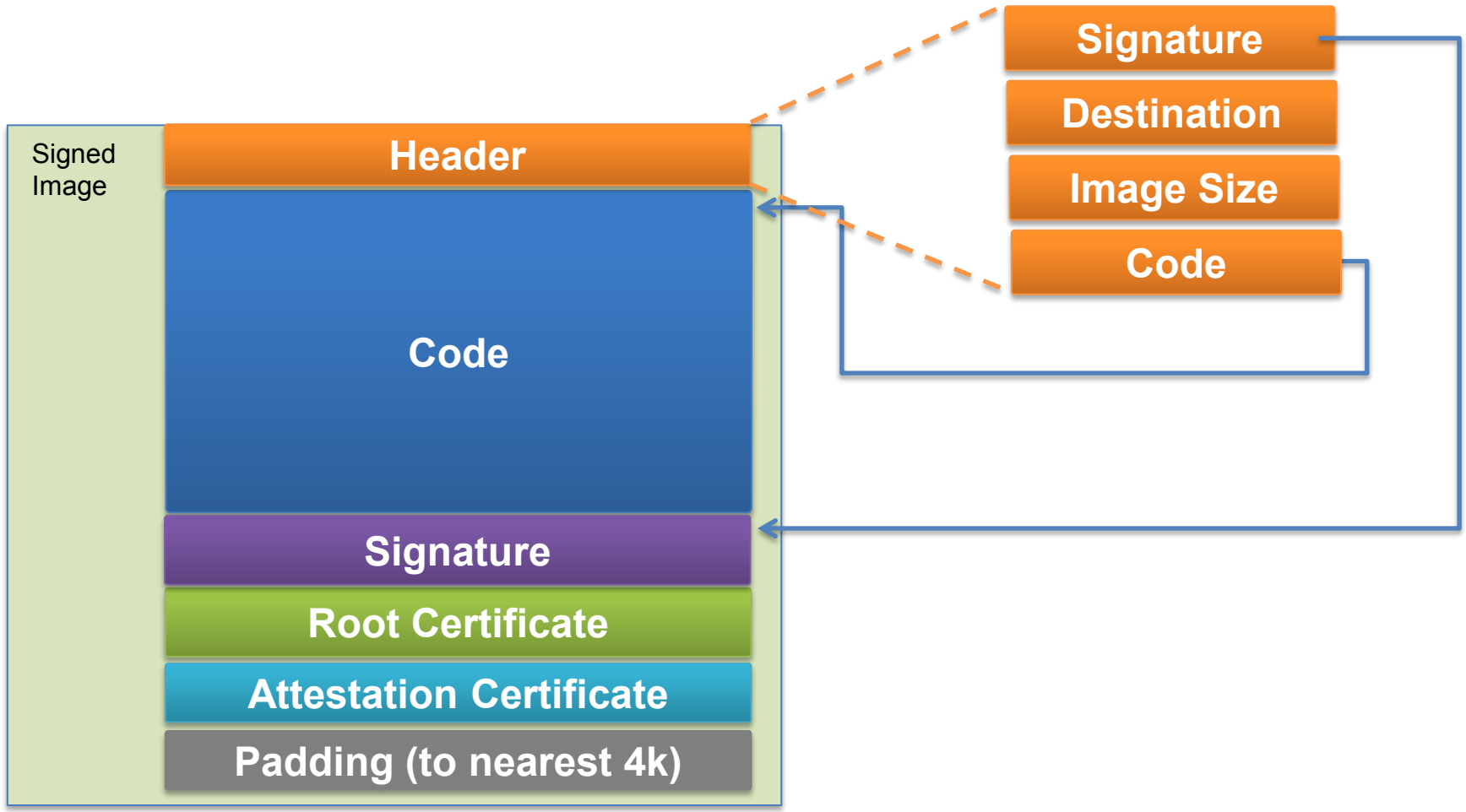
- **Qualcomm SecImage tools**
  - Developed in Python®
  - Can sign images in a chain
    - SBL1, LK, Linux kernel
  - Signature includes code signature and the certificate chain
    - Can consist of two or three X.509 certificates
      - Root certificate
      - Attestation certificate



- **Open Source signLK**
  - signLK creates “dummy” signatures for LK integrity check of non-secure devices
  - signLK cannot be used for provisioning real keys

Structure of a Signed Image

- Layout



- Sign command

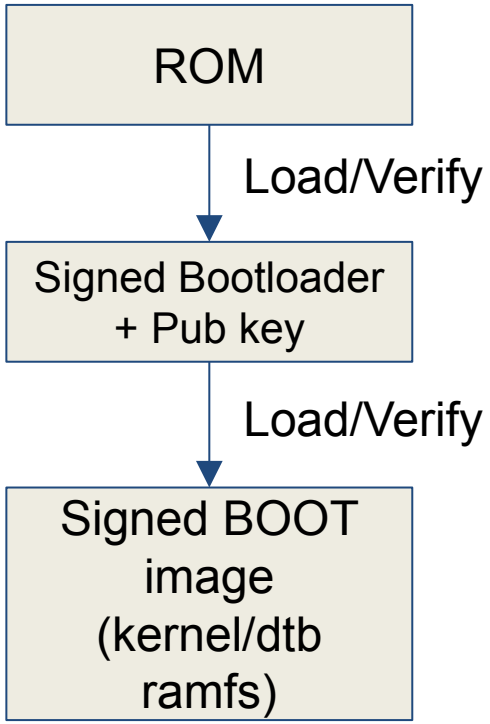
```
$ sectools.py secimage -i <image to sign> -o ./dragonboard-410c/ -g appsbl -c secimage.xml
```

Root Key Deployment

- **QFPROM — Qualcomm® Fuse-Programmable Read-Only Memory**
  - It is Memory made out of multiple 64-bit rows
  - It can be reprogrammed if special read/write fuses are not blown
- **Getting values for the fuses**
  - Use FuseBlower tool, which comes with Sectools
    - Program written in Python
    - Generates sec.dat file based on the public keys used in the signing process
      - Can be flashed into device in a single step
- **Flashing the fuses**
  - Use fastboot command for the initial step
    - Flash sec.dat file in the sec partition of the emmc
  - Upon reboot, SBL1 reads the “sec” partition and loads it into DDR
    - Secure process checks if fuses have been already blown, and if not, it flashes them one by one based on the DDR contents.

DragonBoard 410c Boot Image (kernel, dtb and/or ramfs)

- **"Boot" image**
  - Consists of multiple images combined into one
    - Linux kernel
    - DTB
    - RAMFS
- **Verified bootloader checks boot image signature**
- **Advantages**
  - Standard boot image for the DragonBoard
  - Integrated in OpenEmbedded RPB BSP
  - Low impact on boot time
- **Disadvantages**
  - Limited to RAMFS (read only / size limited by RAM)



LAB — Secure Boot Implementation Example

- Sign and deploy secured image on the DragonBoard 410c

dm-verity (RFS)

- Used in Android™, Chrome OS™
- Operates at block level
  - Below file-system layer
- Uses hash table
- Root hash signed for verification
- Signing key stored in init ramfs
- Advantage
  - Runtime check, minimal boot time overhead, scales well with size
- Drawbacks
  - Read-only RFS
  - No integration with build systems (outside of Android/Chrome OS)
  - Requires block devices

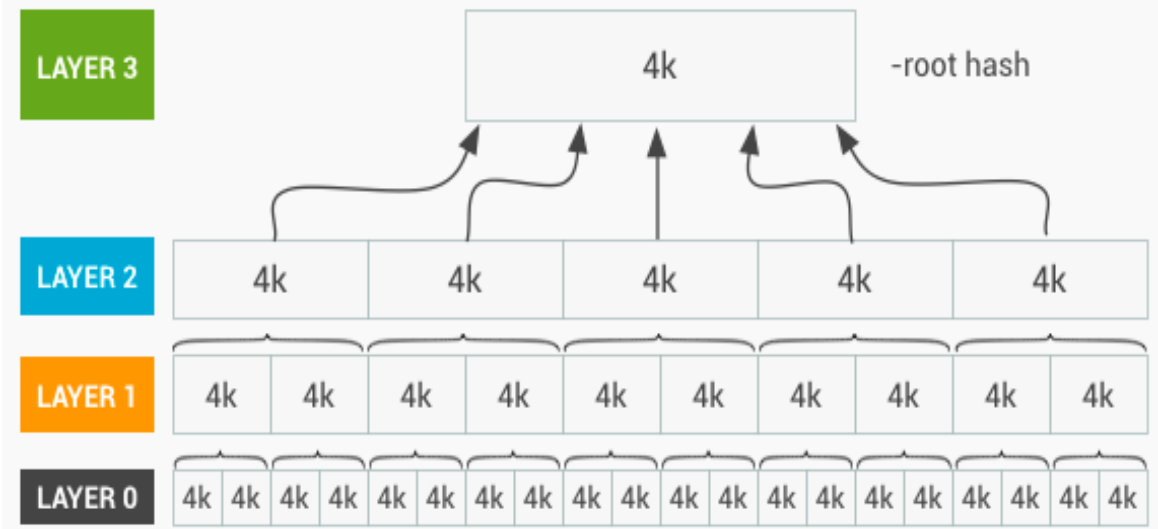
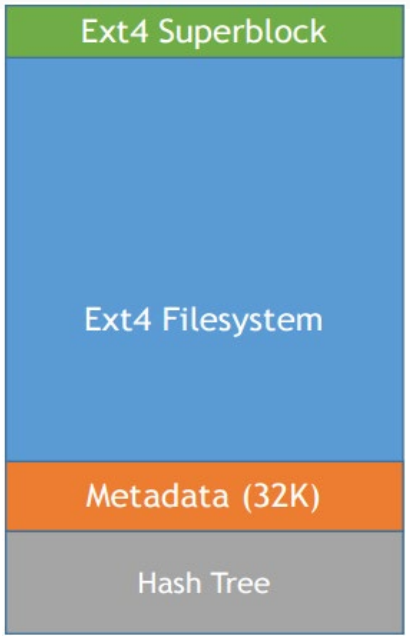


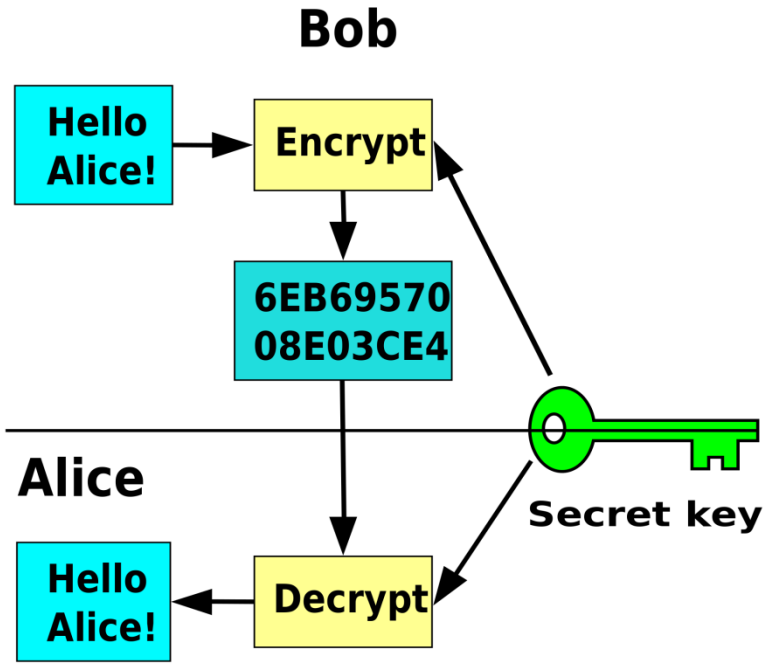
Figure 1. dm-verity hash table



Encryption — Protecting Access

Secure Boot + Encryption

- **Uses symmetric key cryptography**
  - Same key used for encryption and decryption
- **Provides**
  - Confidentiality
  - IP Protection (anti-cloning)
    - Key needs to be unique per device
- **Identify what to protect**
  - Bootloader, kernel, RFS, select applications?
  - Affects firmware update design

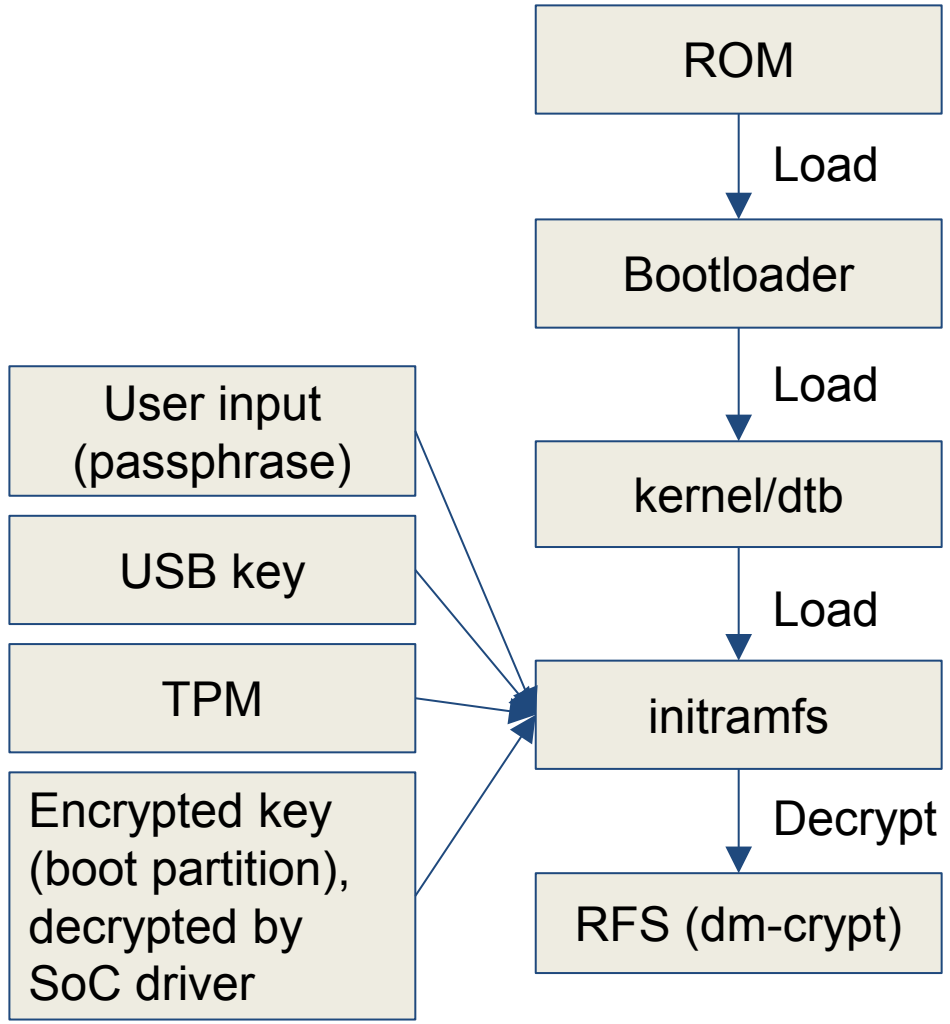


How do we encrypt the system?

Encrypting Data Storage — dm-crypt

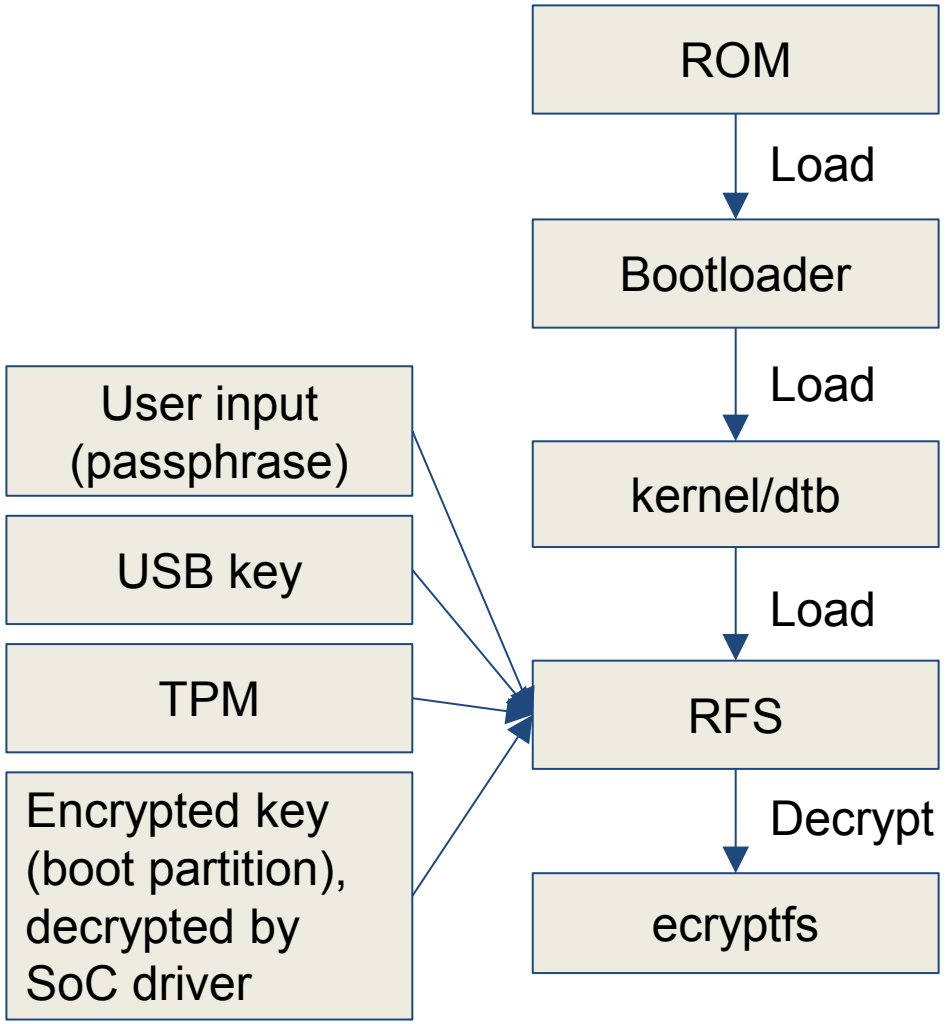
- **Block level**
- **Option for RFS encryption or partitions**
  - Key stored outside RFS
- **Supported on all major Linux distros (Debian®, Ubuntu®) and in Android**
- **Easy setup**
- **Key management on embedded system tricky**
  - Needs a unique hardware ID/key

https://www.timesys.com/security/secure-boot-encrypted-data-storage



Encrypting Data Storage — ecryptfs

- Directory level encryption
- Layers on top of underlying filesystem
- Encryption handled in kernel
- Can use different keys per file
- Supported in all major Linux distros (Debian, Ubuntu) and in Chrome OS
- Remote attestation

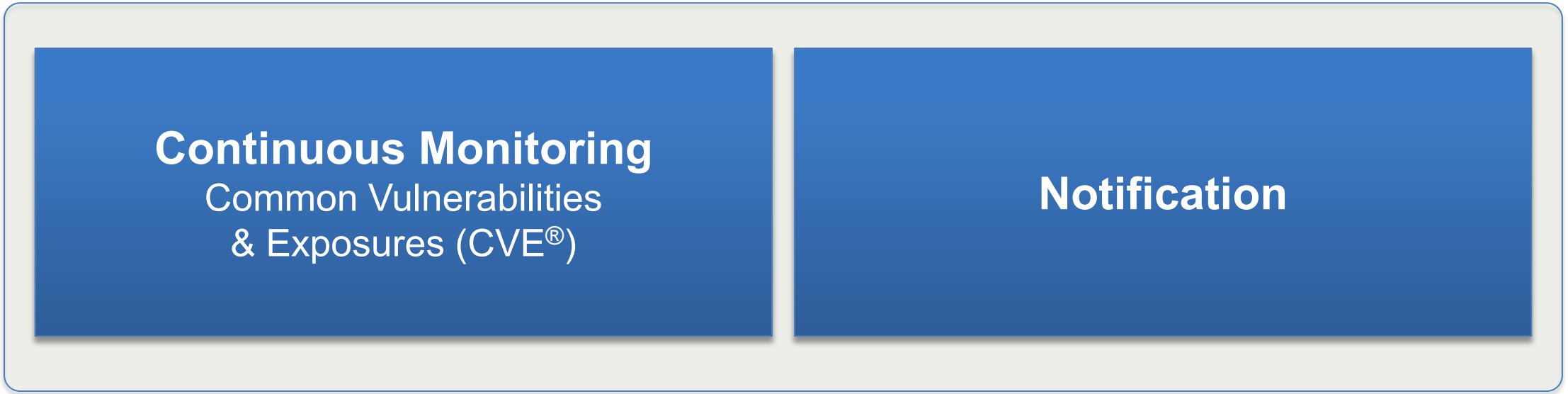


Security Best Practices — Review

- **Disable weak protocols, cryptographic algorithms and/or key lengths**
- **Prune Certificate Authority (CA) list to few trusted**
- **Minimize packages, features in a package**
- **User and/or process permission review**
  - SELinux/MAC
  - Disable root access over ssh
  - Disable serial console access
- **Limit access to signing keys**
- **Review security config of any s/w that opens an external port**
  - Eg: Disable password based ssh, run on non standard port etc.
  - Eg: Disable querying time on ntpd etc.
- **Allow only approved hardware**
  - Disable automount of USBs based on VID, etc.
- **Remove all remnants of the development process (NFS, etc.)**
- **Remove development and/or manufacturing keys from production units**

Ongoing Security

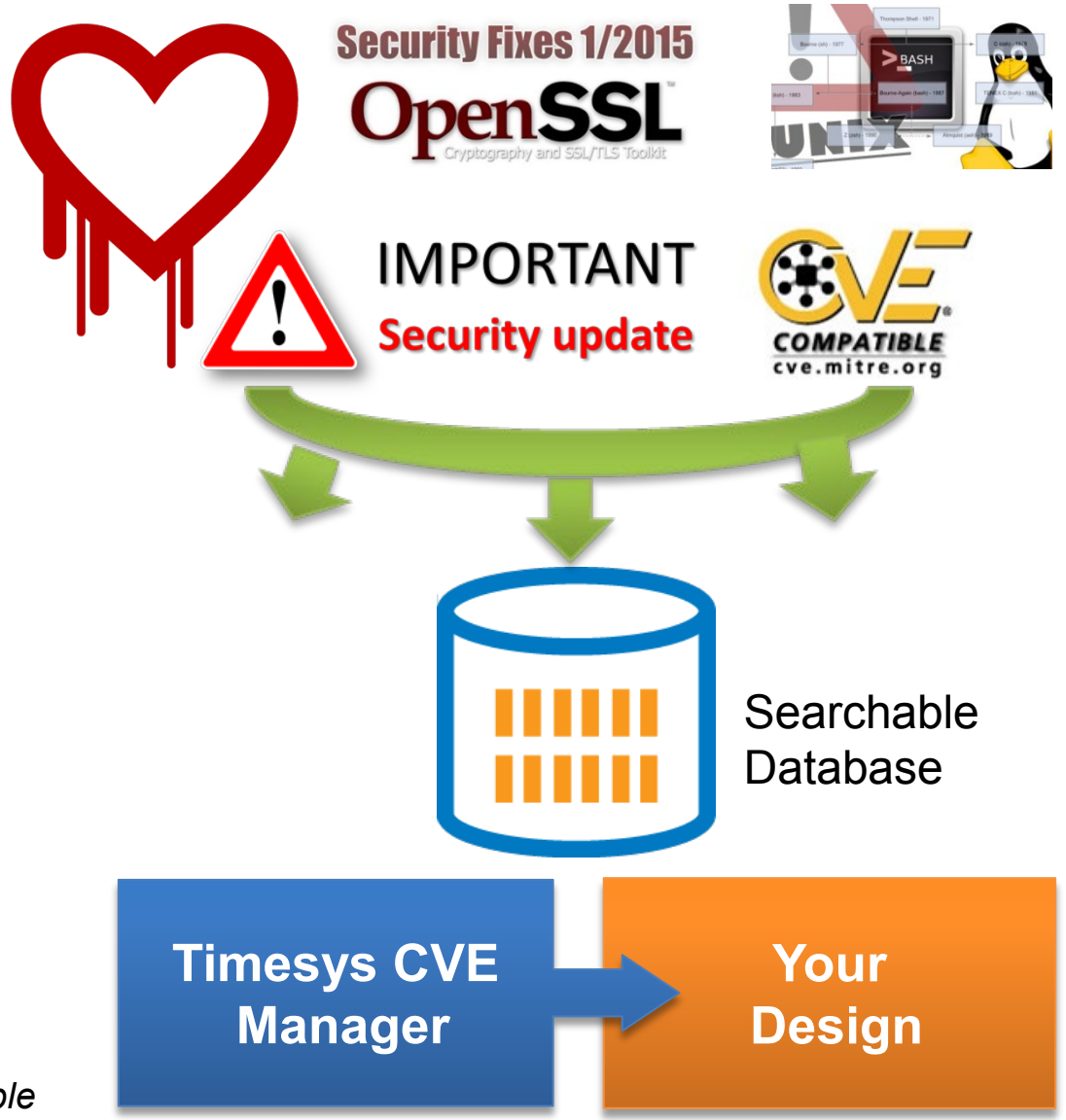
Security Vulnerability Notification Service



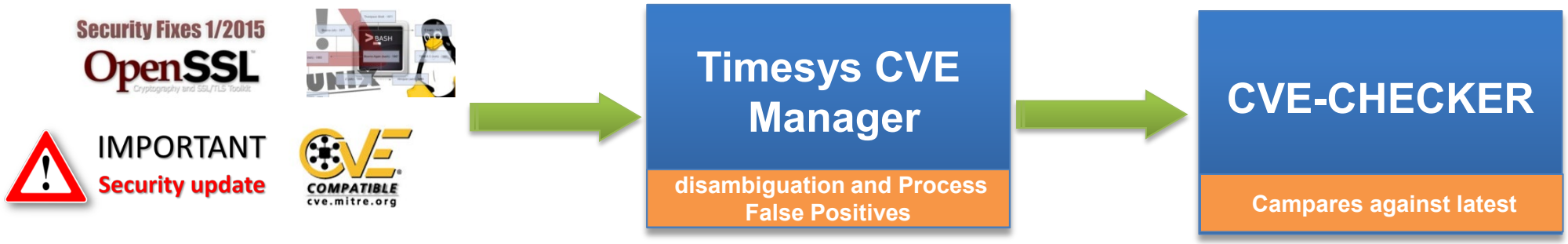
Security is an ongoing process and is not fool-proof. Timesys' security offering provides assistance with minimizing known vulnerabilities based on known issues, but doesn't provide any warranty.

Security Notification Service

- **Common Vulnerabilities and Exposures (CVE) Manager**
  - Tracks security issues from multiple sources
- **Check against your specific software platform (manifest) and notify**
  - Always relevant
- **Differentiate between Unfixed and Fixed**



CVE Manager

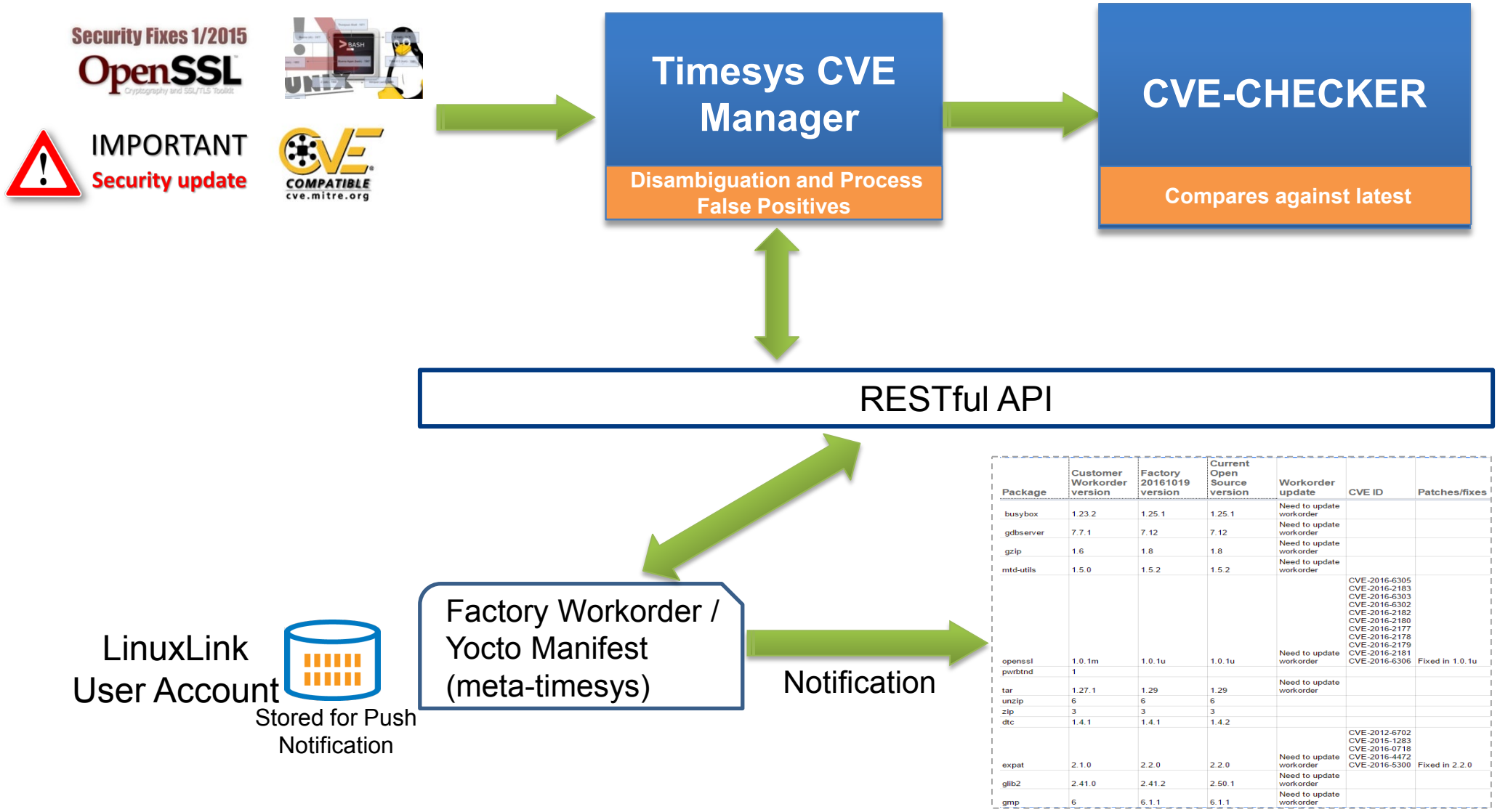


- **Disambiguation**
  - Package names in the CVE database are not exactly the same as in the distributions (Differences between Factory™, Yocto Project® and upstream naming)
- **Only keep relevant**
  - Easy to discern by analyzing the summary
  - Filters out irrelevant CVEs (Oracle® DB, Windows®)
- **False Positives**
  - Possibly relevant but has to be analyzed by an engineer in details
  - E.g. Searching Perl might bring up a CVE but the vulnerability is in a library/module used by the Perl script (marked as a false positive)

CVE Manager Sources

- cves (Common Vulnerabilities and Exposure items) – source NVD NIST
- cpe (Common Platform Enumeration items) – source NVD NIST
- vendor (Official Vendor Statements on CVE Vulnerabilities) – source NVD NIST
- cwe (Common Weakness Enumeration items) – source NVD NIST
- capec (Common Attack Pattern Enumeration and Classification) – source NVD NIST
- ranking (ranking rules per group) – local cve-search
- d2sec (Exploitation reference from D2 Elliot Web Exploitation Framework) – source d2sec.com
- vFeed (cross-references to CVE ids (e.g. OVAL, OpenVAS, ...)) – source vFeed
- ms – (Microsoft Bulletin (Security Vulnerabilities and Bulletin)) – source Microsoft
- exploitdb (Offensive Security – Exploit Database) – source offensive security
- info (metadata of each collection like last-modified) – local cve-search

CVE Manager Notification (Push and Pull)



Package	Customer Workorder version	Factory 20161019 version	Current Open Source version	Workorder update	CVE ID	Patches/fixes
busybox	1.23.2	1.25.1	1.25.1	Need to update workorder		
gdbserver	7.7.1	7.12	7.12	Need to update workorder		
gzip	1.6	1.8	1.8	Need to update workorder		
mtd-utils	1.5.0	1.5.2	1.5.2	Need to update workorder		
openssl	1.0.1m	1.0.1u	1.0.1u	Need to update workorder	CVE-2016-6305 CVE-2016-2183 CVE-2016-6303 CVE-2016-6302 CVE-2016-2182 CVE-2016-2180 CVE-2016-2177 CVE-2016-2178 CVE-2016-2179 CVE-2016-2181 CVE-2016-6306	Fixed in 1.0.1u
pwrbtnd	1					
tar	1.27.1	1.29	1.29	Need to update workorder		
unzip	6	6	6			
zip	3	3	3			
dtc	1.4.1	1.4.1	1.4.2			
expat	2.1.0	2.2.0	2.2.0	Need to update workorder	CVE-2012-6702 CVE-2015-1283 CVE-2016-0718 CVE-2016-4472 CVE-2016-5300	Fixed in 2.2.0
glib2	2.41.0	2.41.2	2.50.1	Need to update workorder		
gmp	6	6.1.1	6.1.1	Need to update workorder		

Ongoing Security

meta-timesys

- **Ongoing security for a product developed with OpenEmbedded RPB BSP is enabled via Timesys provided meta-timesys layer**
  - Works with any BitBake based BSP build system
  - Enables security features through LinuxLink account authentication
  - Uses secure communication with Timesys servers
  - Security provided on specific BSP image (configuration)
    - No security feed for package recipes present in RPB BSP but not used in the product
- **How to run it:**
  - Step 1: Setup BitBake run shell
  - Step 2: From within build directory run the following command

```
$ ../layers/meta-timesys/scripts/manifest.sh rpb-console-image manifest.json
```

  - Step 3: Look at the report. Check for security issues

```
$ ../layers/meta-timesys/scripts/checkcves.py ./manifest.json
```

  - Step 4: Check for security updates in the returned file

Software Manifest Example


```
    },
    "xtrans": {
      "branch": "HEAD",
      "layer": "meta",
      "version": "1.3.5"
    },
    "xz": {
      "branch": "HEAD",
      "layer": "meta",
      "version": "5.2.2"
    },
    "zlib": {
      "branch": "HEAD",
      "layer": "meta",
      "version": "1.2.8"
    }
  },
  "patched_cves": {
    "CVE-2012-2677": [
      "/home/tsu/LAB-410c/96boards-yocto/build-rpb/conf/../../layers/openembedded-core/meta/recipes-support/boost/boost/boost-CVE-2012-2677.patch"
    ],
    "CVE-2015-3310": [
      "/home/tsu/LAB-410c/96boards-yocto/build-rpb/conf/../../layers/openembedded-core/meta/recipes-connectivity/ppp/ppp/fix-CVE-2015-3310.patch"
    ],
    "CVE-2015-7236": [
      "/home/tsu/LAB-410c/96boards-yocto/build-rpb/conf/../../layers/openembedded-core/meta/recipes-extended/rpcbind/rpcbind/cve-2015-7236.patch"
    ],
    "CVE-2015-8607": [
      "/home/tsu/LAB-410c/96boards-yocto/build-rpb/conf/../../layers/openembedded-core/meta/recipes-devtools/perl/perl/perl-fix-CVE-2015-8607.patch"
    ],
    "CVE-2016-1000110": [
      "/home/tsu/LAB-410c/96boards-yocto/build-rpb/conf/../../layers/openembedded-core/meta/recipes-devtools/python/python3/python3-fix-CVE-2016-1000110.patch"
                                                                    2332,6        94%
```

Security Check — Pull Notification Example

```
Requesting image analysis from LinuxLink ...

Recipe:  shadow
CVE ID:  CVE-2017-12424
URL:     https://nvd.nist.gov/vuln/detail/CVE-2017-12424
CVSS:    7.5
Status:  Unfixed

Recipe:  util-linux
CVE ID:  CVE-2015-5224
URL:     https://nvd.nist.gov/vuln/detail/CVE-2015-5224
CVSS:    7.5
Status:  Early Notice

Recipe:  libtirpc
CVE ID:  CVE-2017-8779
URL:     https://nvd.nist.gov/vuln/detail/CVE-2017-8779
CVSS:    7.8
Status:  Unfixed

Recipe:  icu
CVE ID:  CVE-2016-6293
URL:     https://nvd.nist.gov/vuln/detail/CVE-2016-6293
CVSS:    7.5
Status:  Unfixed

Recipe:  openssl
CVE ID:  CVE-2016-7055
URL:     https://nvd.nist.gov/vuln/detail/CVE-2016-7055
CVSS:    2.6
Status:  Fixed
Patched by:
        /home/tsu/LAB-410c/96boards-yocto/build-rpb/conf/../../layers/meta-timesys/manifest/meta-openembedded/meta-
networking/recipes-connectivity/openssl/openssl/CVE-2016-7055.patch

Recipe:  gnutls
CVE ID:  CVE-2017-5336
URL:     https://nvd.nist.gov/vuln/detail/CVE-2017-5336
CVSS:    7.5
Status:  Unfixed
```

LAB — Ongoing security

- Generate a software manifest
- Submit the software manifest for security check

Timesys Security Offering Summary

Security Notification Subscription

Continuous Monitoring
Common Vulnerabilities & Exposures (CVE)

Notification
Push and Pull

Stay Secure Service

Patching
BSP Lifecycle Maintenance

Deployment
High Assurance Boot

Secure By Design Service

Security Audit and Scanning

Security Hardening

Security is an ongoing process and is not fool-proof. Timesys' security offering provides assistance with minimizing known vulnerabilities based on known issues, but doesn't provide any warranty.

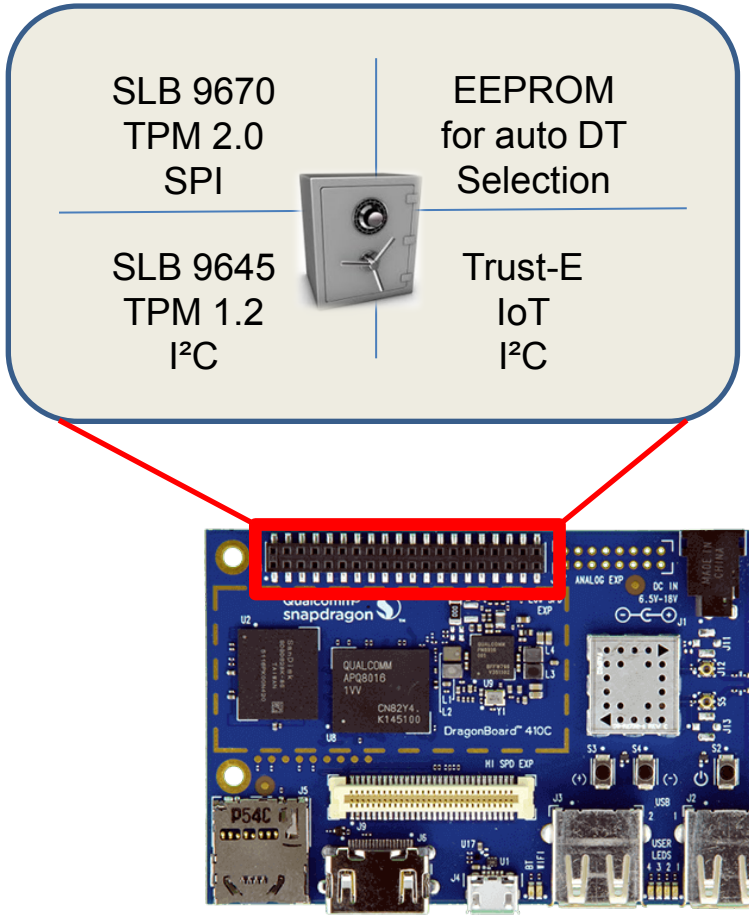
Trusted Platform Module (TPM)

What Is a Trusted Platform Module (TPM)?

- **Mechanism to verify the integrity of remote clients/servers**
  - Correct/Authenticated software is installed
  - Verifies that system has not been compromised
  - Authenticates services
- **Standard defined by the Trusted Computing Group**
- **Can be mounted/used in addition to the security mechanisms available on the main CPU and can store**
  - Keys
  - Certificates
- **Available as an add-on 96Boards™ module for the DragonBoard 410c**
  - TRESOR

TRESOR

- ## Concept



OPTIGA™ TPM Security Functions

Device Authentication
- One-way authentication
- Mutual authentication

System integrity
- Secure Boot
- Remote platform verification

Secure Channel
- Encrypted Communication
- Key Generation

Dedicated functions for
- Platform manufacturer
- System operators
- Vendor/User/Enterprises

User Management
- Password Protection
- User management and keys

Lifecycle Management
- Key Backup and refurbishment
- Personalization and identities
- Supply chain tracking

Secure Updates
- Remote maintenance
- In-field flexibility and reaction

Secure Clock and Time
- Reliable clock when offline
- Timer and Monotonic Counter

TRESOR Features

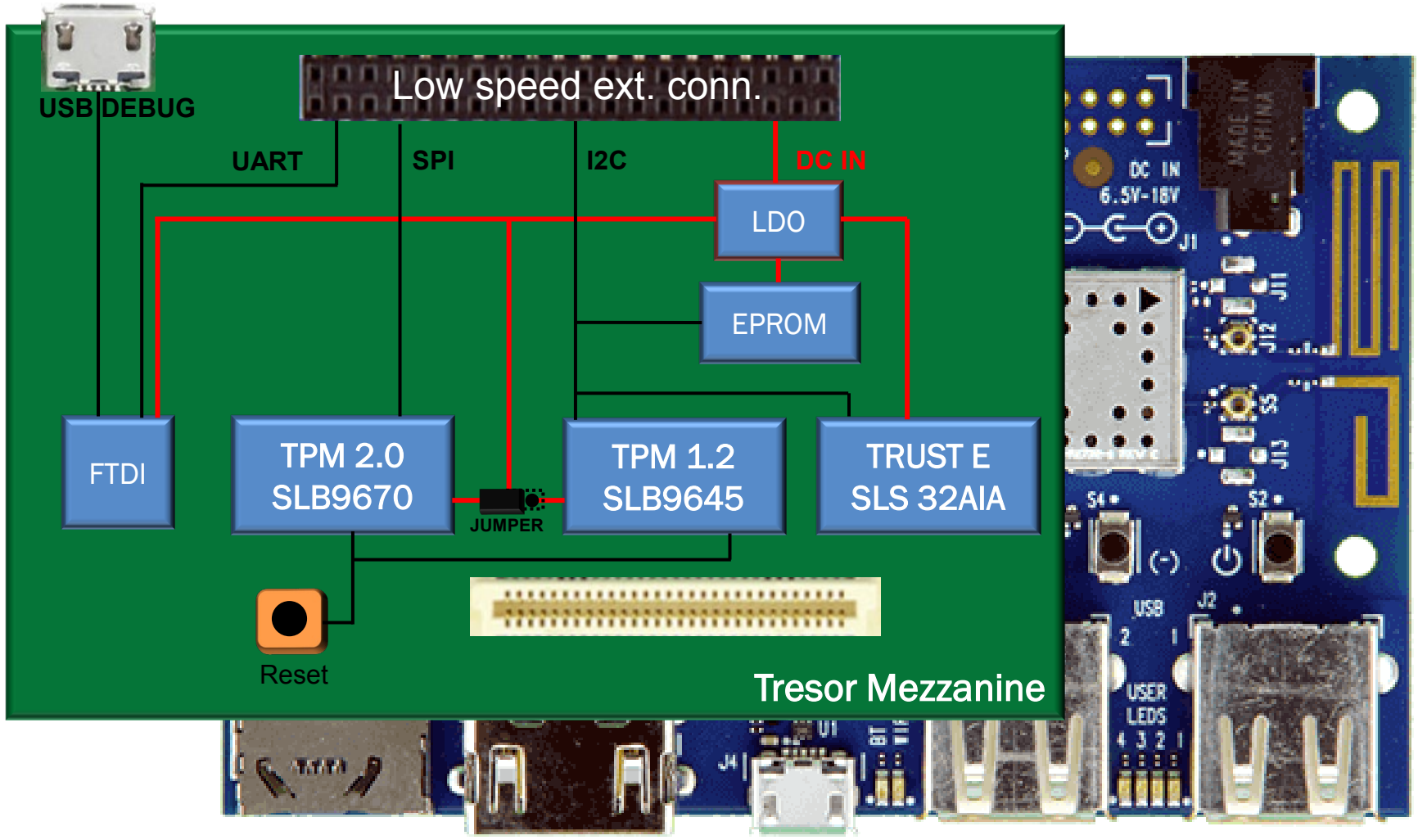
- **Complete Security Verification Toolkit**
  - Tresor gives the developers and the maker community access to latest security solutions — both for evaluation but also for own developments

  - Features:
    - TPM 1.2
    - TPM 2.0
    - Lightweigt security solution based on Trust-X Technology (added in V2 of the board after release of Trust-X)
    - EEPROM to enable auto Device-Tree selection feature

  - Software:
    - TPM Framework in Linux kernel
    - TrouSers
    - Hardened OpenSSL / GNUTLS

TRESOR Block Diagram

- **Complete Security Verification Toolkit**



Software Enablement

- **The use of the TRESOR TPM module requires additional software**
  - Linux kernel device driver for the TPM
  - TPM utilities
- **Device driver is TPM module specific**
  - Available for the 96Boards module, present in the Linux kernel source tree
  - Requires Linux kernel reconfiguration to enable
  - Uses additional definitions in the DTB file
- **User space enablement**
  - Requires additional packages to be installed in the RFS
    - tpm-tools
    - trousers
    - openssl_tpm_engine
  - Recipes available for both, provided by the meta-security metalayer
- **Once all software is integrated, RPB BSP images must be generated**

Example Use of TPM — SSL Certification

- **SSL is used in every secured channel between server and client in the Internet connection**

- **TPM protects the keys**
  - The key can only be decrypted and used inside the TPM
  - It is always protected
- **Server TPM stored key is used to authenticate itself**
  - If a key could be obtained by a hacker, it could be used to authenticate itself against clients as a valid server, hijacking this way client connections and possibly collecting secure data!!!

Questions?

Source code, deployment images and SDK can be downloaded from linuxlink.timesys.com

---

developer.qualcomm.com 96boards.org arrow.com timesys.com

Thank you

Follow us on:

For more information, visit us at:
www.qualcomm.com & www.qualcomm.com/blog