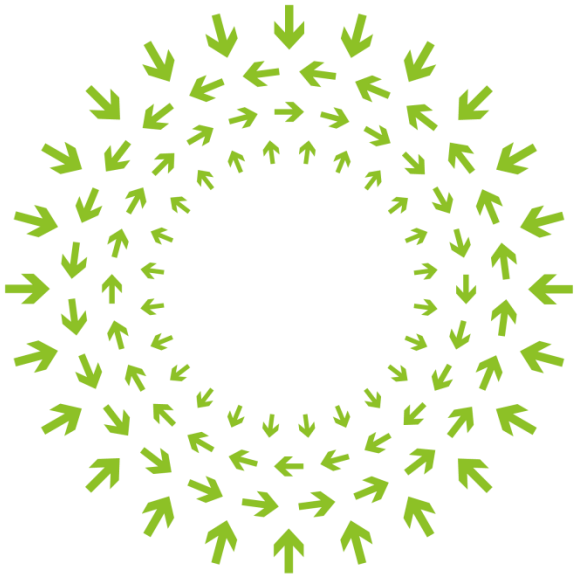

OPEN
Compute Project

Project Cerberus
Root of Trust Specification

Author:
Bryan Kelly, Principle Firmware Engineering Manager, Microsoft

Revision History

Date	Description
8/28/2017	Version 0.1 – Initial Draft

As of November 1, 2017, the following persons or entities have made this Specification available under the Open Web Foundation Final Specification Agreement (OWFa 1.0), which is available at http://www.openwebfoundation.org/legal/the-owf-1-0-agreements/owfa-1-0

Microsoft Corporation.

You can review the signed copies of the Open Web Foundation Agreement Version 1.0 for this Specification at http://www.opencompute.org/participate/legal-documents/, which may also include additional parties to those listed above.

Your use of this Specification may be subject to other third party rights. THIS SPECIFICATION IS PROVIDED "AS IS." The contributors expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to the Specification. The entire risk as to implementing or otherwise using the Specification is assumed by the Specification implementer and user. IN NO EVENT WILL ANY PARTY BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

CONTRIBUTORS AND LICENSORS OF THIS SPECIFICATION MAY HAVE MENTIONED CERTAIN TECHNOLOGIES THAT ARE MERELY REFERENCED WITHIN THIS SPECIFICATION AND NOT LICENSED UNDER THE OWF CLA OR OWFa. THE FOLLOWING IS A LIST OF MERELY REFERENCED TECHNOLOGY: INTELLIGENT PLATFORM MANAGEMENT INTERFACE (IPMI); I^2C IS A TRADEMARK AND TECHNOLOGY OF NXP SEMICONDUCTORS ; EPYC IS A TRADEMARK AND TECHNOLOGY OF ADVANCED MICRO DEVICES INC.; ASPEED AST 2400/2500 FAMILY PROCESSORS IS A TECHNOLOGY OF ASPEED TECHNOLOGY INC.; MOLEX NANOPITCH, NANO PICOBLADE, AND MINI-FIT JR AND ASSOCIATED CONNECTORS ARE TRADEMARKS AND TECHNOLOGIES OF MOLEX LLC; WINBOND IS A TRADEMARK OF WINBOND ELECTRONICS CORPORATION; NVLINK IS A TECHNOLOGY OF NVIDIA; INTEL XEON SCALABLE PROCESSORS, INTEL QUICKASSIST TECHNOLOGY, INTEL HYPER-THREADING TECHNOLOGY, ENHANCED INTEL SPEEDSTEP TECHNOLOGY, INTEL VIRTUALIZATION TECHNOLOGY, INTEL SERVER PLATFORM SERVICES, INTEL MANAGABILITY ENGINE, AND INTEL TRUSTED EXECUTION TECHNOLOGY ARE TRADEMARKS AND TECHNOLOGIES OF INTEL CORPORATION; SITARA ARM CORTEX-A9 PROCESSOR IS A TRADEMARK AND TECHNOLOGY OF TEXAS INSTRUMENTS; GUIDE PINS FROM PENCOM; BATTERIES FROM PANASONIC. IMPLEMENTATION OF THESE TECHNOLOGIES MAY BE SUBJECT TO THEIR OWN LEGAL TERMS.

Contents

List of Figures

1 Introduction

This document will describe to hardware capabilities of the Cerberus RoT.

2 Processor Features

The Cerberus RoT is a Micro Control Unit (MCU) that interposes on the SPI/QSPI/eSPI bus. Internally, the RoT will contain 256KB or greater of SRAM. The RoT will interpose on the SPI bus between the platform or component processor and its SPI flash. The RoT will support the following functionality:

- Secure boot.
    - RSA with SHA256 for authenticated image boot.
    - AES256 in Galois Counter Mode (GCM) for encrypted image boot.
- DMA controller, configurable all memories and DMA-capable peripherals. Primarily for bridging SPI interface memory buffer to SPI interface memory buffer.
- CRC engine block for calculating CRCs using 3 standard polynomials.
- 32-bit general-purpose timers/counter for clock synchronization.
- DICE Architecture for establishing Root of Trust.

2.1 Connectivity

- 2 x Serial interfaces that can be configured as slave and operate at 48Mhz or higher.
- 2 x I2C-bus interfaces that support Fast-mode and Fast-mode Plus with data rates up to 1 Mbps. Two sets of I2C interfaces should support both master and slave modes.
- eSPI interface with eSPI slave functionality. Configurable to operate as eSPI slave and support Slave Attached Flash Sharing (SAFS).
- QSPI Master Interface with chip select for two chips.
- QSPI slave should also be possible on the SPI interfaces.
- General-Purpose Input/Output (GPIO) pins selectable as interrupt pins

  .

2.2 Security

- AES hardware engine supporting 128, 192, and 256 bit keys.
- SHA2 256-bit hardware engine.
- Physical Unclonable Function (PUF) for protecting local key stores of up to 4096-bit keys.
- RFC4122 version 5 compliant 128-bit UUID for each device
- FIPS 140-1 compliant Random Number Generator (RNG)
- OTP memory for storing root keys.
- OTP memory for key revocation
- DICE Architecture
- Multi stage bootloader, support key revocation and RIoT security implementation

3 Security Details

The security system on the Cerberus RoT consists of hardware blocks for AES encryption, SHA hashing, random number generation, and key storage. All functions are accessible via the processor and DMA engine for data encryption/decryption and hashing.

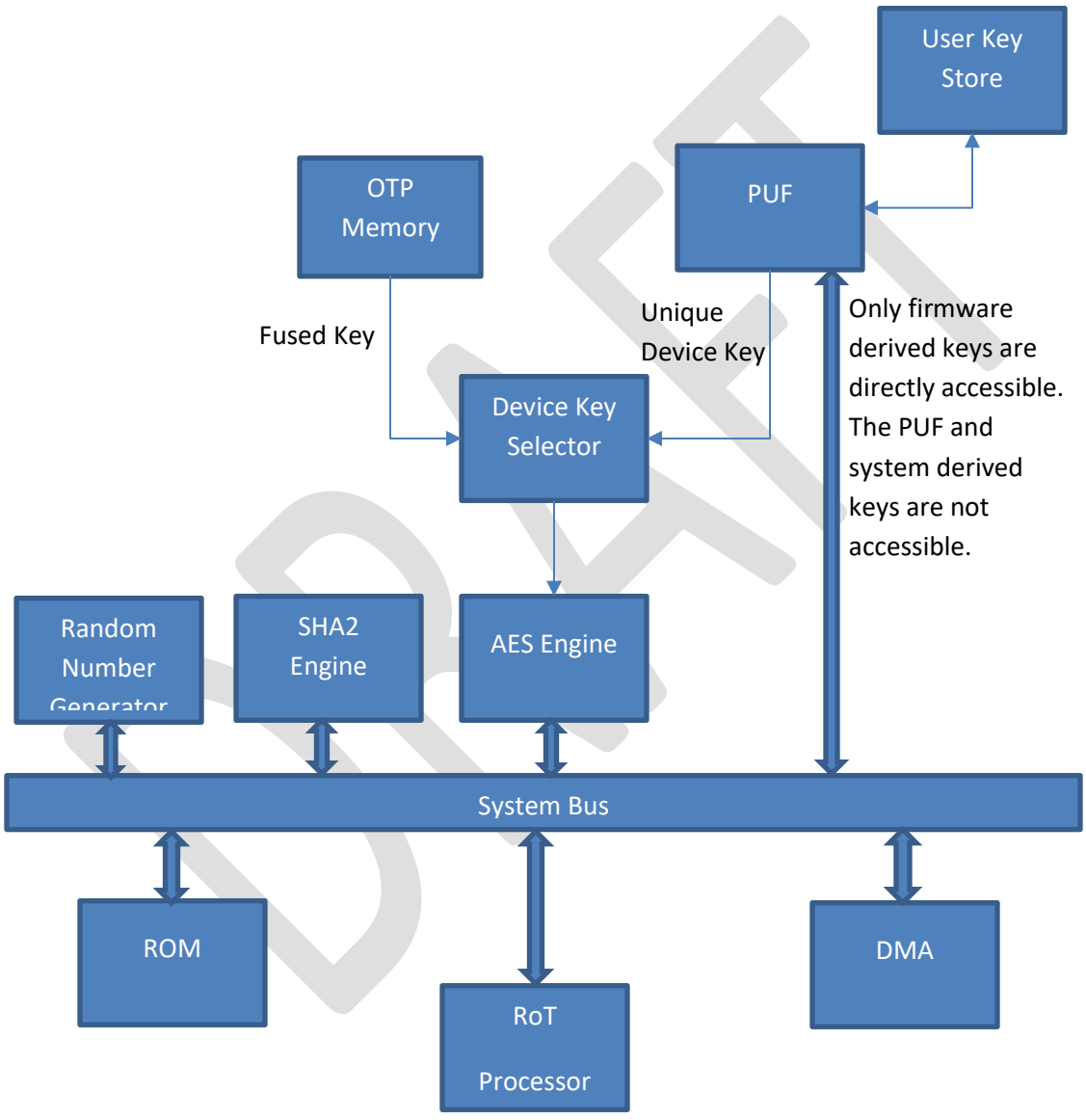

Figure 1 Cerberus RoT Security System

3.1 Key Storage

Key storage is achieved through one-time programmable (OTP) memory and an SRAM based physical unclonable function (PUF). The key store is connected to the AES engine to provide protection for the most critical keys that are not accessible directly by the software.

Keys stored in OTP can be loaded directly into the AES engine for encryption/decryption. Also, a 256-bit key can be descrambled prior to use by the AES engine.

The PUF provides a unique key per device that can be used for boot loading or encrypting critical data. Additional keys can be stored by the PUF and read by software. These keys will be scrambled with the unique device ID prior to storing.

3.2 AES Engine

The AES engine provides hardware acceleration for encrypting and decrypting data at boot and during run-time. The engine supports using key sizes of 128, 192, and 256 bits in the following modes:

- Electronic Codebook (ECB)
- Cipher Block Chaining (CBC)
- Cipher Feedback (CFB)
- Output Feedback (OFB)
- Counter (CTR)
- Galois/Counter Mode (GCM)

In addition to the accessing the secure keys from OTP or the PUF, software supplied keys can be provided to the AES engine. These can be additional keys stored in the PUF or stored by some other means in the software.

3.3 SHA Engine

The SHA engine provides hardware support for generating 256-bit SHA2 hashes of data. The hashing engine can be used with the DMA engine to allow complete hashing offload of data.

3.4 Random Number Generator (RNG)

The Cerberus RoT contains a true random number generator that can be used to generate additional random keys, as required. The RNG is FIPS 140-1/2, NIST, DieHard compliant.

3.5 Physical Unclonable Function (PUF)

The PUF present on the Cerberus RoT provides each device with a unique key that is used to secure other aspects of the system as well as provide a secure way to store additional keys. The device key is not directly accessible via software, but can be used with the AES engine to encrypt critical data in a device-specific way with a completely private key build into the part.

The PUF can storing keys of 64 to 4096 bits. These additional keys are available to software for general use. When stored, these keys will be scrambled with a unique device ID in the same way as the scrambled OTP key.

4 Secure Boot

In order to provide a valid root-of-trust, the Cerberus RoT itself needs to be trusted. This is enforced by requiring the RoT to only run verified software via the secure boot ROM code in the RoT. There are two modes of secure boot supported by the RoT:

- Image signing which provides authentication only.
- Image encryption which provides obfuscation, and in GCM tag mode include authentication.

Both secure boot methods rely on a hardware root-of-trust based on the key fused into the OTP memory on the RoT, both methods can be combined to provide authentication and encryption.

4.1 Signed Image Booting

When using secure boot with signed images, only images signed with the private RSA key that matches the public key fused into the OTP memory will be loaded.

The boot process begins with the ROM bootloader reading the signed firmware image header. This image header will contain the signature of the image and the public key corresponding to the private key used to generate the signature. The signature is a SHA256 hash of the image RSA encrypted with a 2048-bit key.

Once the image header is read, the public key is extracted and a SHA256 hash is generated for the key. This hash is compared to the hash of the public key fused in the RoT OTP memory. If these hashes match, the image was signed with the correct key, so boot can proceed.

After validating the key used to the sign the image, the image itself is validated. A SHA256 hash is generated for the image data and compared to the signature that is decrypted with included public key. If the hashes match, this image is known to be valid and will be loaded.

Any further authentication is handled by the loaded firmware.

Figure 2 below shows the boot flow when using signed images for authentication.

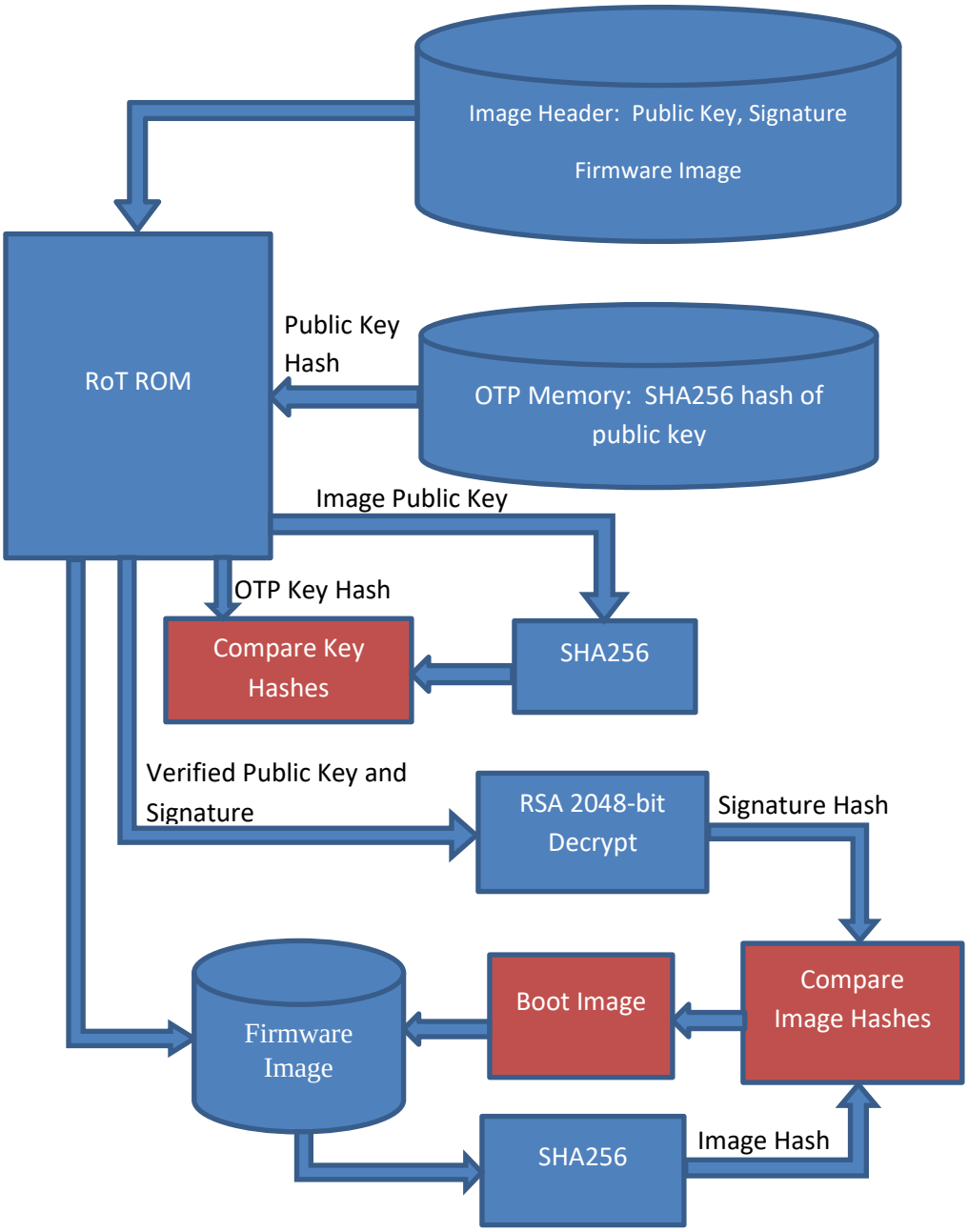

Figure 2 Signed Image Boot Flow

4.2 Encrypted Image Booting

Booting with an encrypted image grants the option of enforcing additional security. In addition to the image being encrypted, a GCM authentication tag can be fused into the RoT to ensure only a single image is every allowed to boot. In this boot mode, images are encrypted using AES-GCM with a 256-bit key.

Just as for signed images, the ROM bootloader begins the boot process by loading the encrypted image. In this case, the image contains the encrypted executable binary and the initialization vector (IV) to use for decryption. If the GCM authentication tag is not fused into the RoT, this tag will also be included with the image.

After loading the image, it is decrypted using AES-GCM with the 256-bit key fused in the RoT. This process will generate the decrypted firmware image and the authentication tag.

If the GCM authentication tag is fused in the RoT, then the tag is compared to the one that is fused. Otherwise, the tag is included as part of the image header for verification. If the tag doesn't match, the image doesn't boot.

Once the firmware is loaded, additional authentication can be executed as necessary.

Figure 3 below shows the boot flow when using encrypted images.

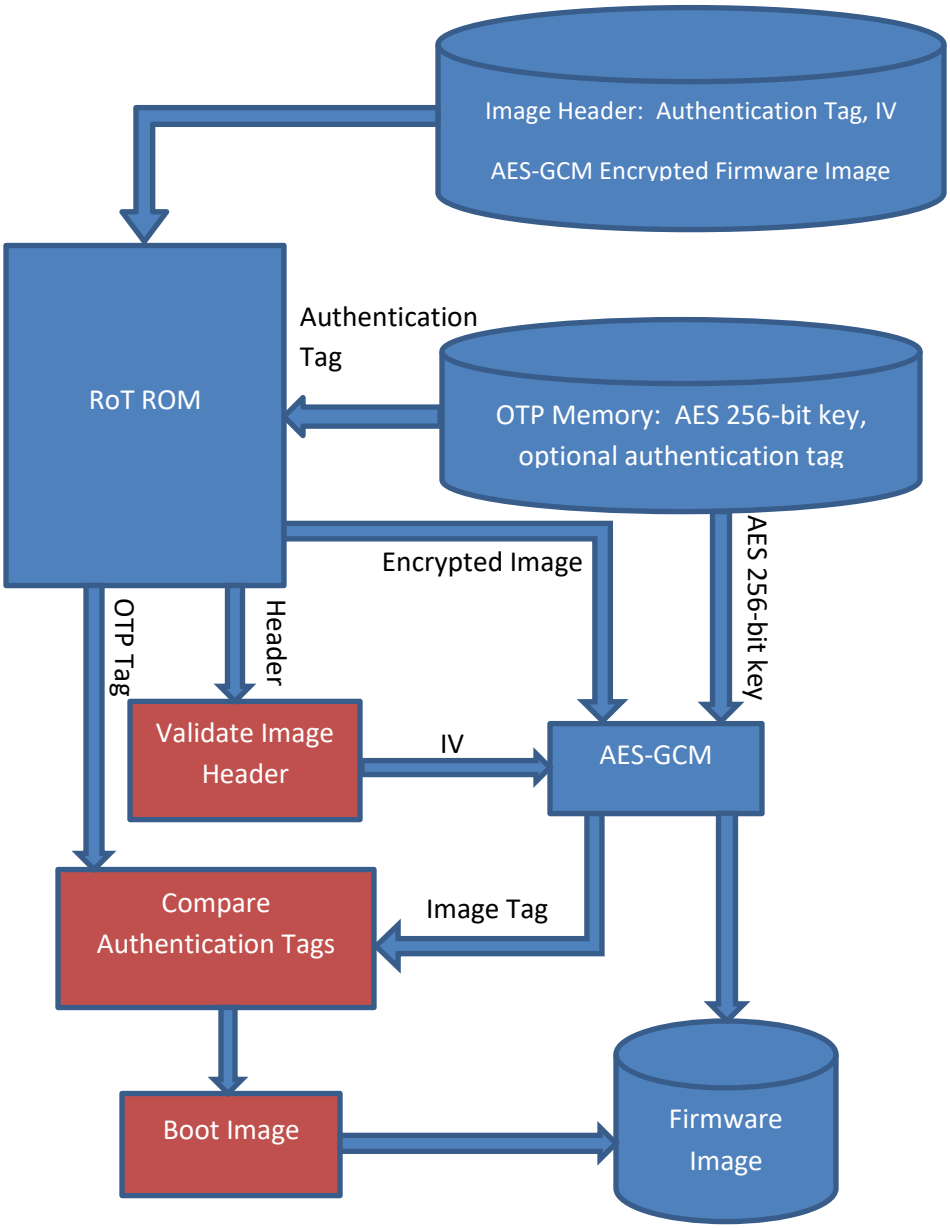

Figure 3 Encrypted Image Boot Flow