

Features

- Multi-phase loader cryptographically validates a phase before loading it
- Supports hardware assist to speed boot time
- Secure access to engineering mode bypasses application loading for lab debugging purposes
- Based on FIPS validated cryptographic algorithms and standards like RSA, ECC and AES
- Low footprint solution
- Binary code or C language source code to facilitate adaptation to the end system
- Builds on ARM, PowerPC, X86 platforms

Applications

- Anti-tampering and anti-cloning
- Design IP protection
- Content protection
- Feature control and decommissioning

Markets

- Mobile phones
- Set top boxes
- Networking
- Computing

Ellipsys Trust Framework

- ESS-04: Ellipsys-SB
- ESS-06: Ellipsys-CA
- ESS-07: Ellipsys-VSM

Secure boot can greatly enhance the security of an embedded system by cryptographically verifying that the code being loaded and executed is authentic and has not been unknowingly or maliciously modified. Once a feature of high security applications, these techniques are now being adopted by a broad range of products such as mobile phones, set-top boxes and networking systems such as base stations, routers and other infrastructure devices.

ESS-04 is part of Ellipsys Trust Framework and it allows developers to implement secure boot systems using software only constructs or to use Elliptic offload engines to simplify and accelerate the required verification or decryption operations in the target system.

General Description

Overview of Ellipsys-SB

Secure boot systems rely on well proven cryptographic algorithms for signing and optionally decrypting code to ensure that the processor is initialized into a known state and is running code from a trusted source.

In Ellipsys-SB, multiple phases are used to segment the bootstrap function, which allows each phase to release system resources to use if that facility exists in the SoC or system architecture.

The multi-phase hierarchy also provides support for multiple, independent trusted code sources which is required should several companies be contributing to a code build.

The phases are outlined below:

Phase 0:

This is a minimal system loader which self-tests the required cryptographic hardware components (if available), then loads and verifies Phase 1. The loader does the cryptographic verification and optional decryption of the Phase 1 loader code. Elliptic offers three options for Phase 0 – an all-hardware solution in a finite-state machine, a firmware-only solution or a combination of both. The choice is based upon the security required and the system cost objectives. Elliptic recommends that the firmware-only solution be implemented with the Phase 0 code stored in ROM to enhance the protection of the embedded system.

Phase 1:

Phase 1 is full-featured loader with cryptographic signature verification and optional code decryption of the final operating system and application code. In SoC designs, it is a software loader which resides in external Flash and is loaded into on-chip SRAM for execution after the Phase 0 loader cryptographically verifies it. If desired, the Phase 1 loader can be decrypted as well to hinder attackers' efforts to reverse-engineer the system. The software loader may also leverage security offload resources if available.

Phase 2:

Phase 2 is the target Operating System and Application. In sophisticated systems such as SoCs for mobile phones, the Application can consist of code from many different developers including the IC vendor, the handset vendor, network provider and even downloaded applications from third parties. It is also possible that multiple feature sets can be implemented in an SoC and selectively made available during the secure boot process based on their contracted service level through a field configuration update. Elliptic can support many key permutations to permit a flexible software development scenario.

General Description cont'd

Ellipsys-SB also supports secure updates. As an update changes the length and signature of the Phase 2 code, the secure update utility must modify and re-write the credentials in a secure manner.

Secure boot relies on the availability of security credentials such as keys or certificates. The nature of these credentials and how they are stored depends on the threat model and target system (SoC, board or system). The final configuration is jointly agreed during the architecture stage.

It is also possible to retrofit an existing design to implement secure boot. This may result in agreed limits to the integrity of initial implementation of secure boot which can then be addressed should the product design be updated to add new features or during a cost reduction. Ellipsys-SB offers sufficient flexibility to address both new architectures as well as existing designs.

Sample Implementation

Figure 1 illustrates an example implementation of secure boot using the Ellipsys-SB. This architecture uses the following hardware and software resources:

- On-chip NVM that contains the RSA Root public key Hash value and Silicon ID for the device
- Flash Memory contains the boot images to be verified by the Secure Boot SDK authentication functions.
- The Boot Application which is the boot reference application that implements 3-phase boot process.
- The Secure Boot SDK implements boot authentication functions.
- Phase 2 or Mission Mode is the final operating system and applications code to run on the target device

Ordering Information

- Ellipsys-SB is highly configurable and it can support many permutations in secure boot. Further technical details on the options available to developers can be provided under NDA.
- Please contact Elliptic directly to conclude an NDA and then receive technical information on the Ellipsys-SB.

