APPLICATION NOTE

Secure Boot Simplified

Using nonvolatile memory to store the operating program for a modern digital system can shorten time-to-market, permit downloading in the field to enhance operation, and allow diversified functionality with the same hardware base platform. However, if an unauthorized program can be loaded into the memory, the operation of the system may be unpredictable. The consequence of this may be damage to the OEM's reputation, liability issues due to system malfunctions, disclosure of protected media, use of the system beyond recommended limits, or other problems.

Modern systems are often built using a standard microprocessor with the operating program stored in flash memory. Such an architecture can permit the manufacturer to quickly take advantage of advances in processor performance and memory cost, while offering fast time to market. If the operating program is stored in an accessible flash memory, it's very hard to prevent an adversary from obtaining its contents and modifying it to run a fraudulent program.

Using standard parts for the board bill of materials also makes it easy for an adversary to just purchase the same components, copy the board, read the contents of the flash memory, and write that image into the flash memory to build a clone system. They can then sell the system without all the development engineering effort. This can be a significant financial liability or apparent quality issue for a manufacturer.

In combination with an external security device, a secure boot procedure can be implemented which addresses these two issues.

Typically, on power-up, the microprocessor begins execution from a boot memory which is usually located on the processor device, often in mask programmed ROM or perhaps in a section of locked flash memory. The boot program stored there may initialize the hardware and external memory interface system before starting execution of the operating program stored in flash.

To implement secure boot, a validating code or signature is stored in flash along with the operating program. As part of the boot program execution at system startup, the operating program is verified by the security device using this signature to ensure that the program is authentic. If the verification succeeds, the operating program is executed and the system operates in the normal way. Any modification of the operating program – even of just a single bit – will require a new validating signature.

Symmetric secure boot procedures using an external security device follow the following flow:

- At the factory, a hash algorithm is used to create a digest of the operating program. This digest is combined with a validating secret to create the signature (Step 1). Both the program and the signature are stored in the system flash memory (Step 2).
- On startup, the boot program calculates the digest of the operating program stored in flash using the same hash algorithm (Step 3). It also reads the signature from the flash memory.
- The boot program transmits the digest and the signature to the security device (Step 4). The security device combines the digest with the secret and compares the result with the signature that was passed to it from the boot program (which came from the flash). The security device passes a yes (comparison succeeded) or a no (input signature not what was expected) back to the processor (Step 5).
- If the validation fails, indicating an unauthorized operating program, the boot program may prompt the user to download a new authentic image, return an error indication, or take other action (Step 6).

The Atmel® ATSHA204 hardware security device from Atmel was specifically designed to store the boot validation secret in a general purpose system. It can resist the most determined logical or physical attack so that the adversary will not be able to determine the validation secret. Since it implements the entire signature generation and checking function in secure hardware, returning only a simple yes or no to the system, an adversary is prevented from using the device to generate the proper signature for a fraudulent program.

An adversary can purchase an ATSHA204 device from Atmel. Without knowing the value of the validating secret, the adversary would not be able to personalize the ATSHA204 to operate properly in the system. As a result, system cloning is now more difficult. Contact Atmel for more information on clone prevention.

Many readers will have additional questions or concerns:

- Isn’t it expensive to modify the ROM boot code on a microprocessor?
- Can’t the adversary just modify the system to include a fake security device which always sends back a yes answer?
- What about using a version of the microprocessor which has a different boot program that doesn’t include secure boot?
- What about other possible hardware modifications to the system to circumvent the secure boot procedure?
- What happens when a new operating program is downloaded to the system? Can every system in the field have the same downloaded program? Can the operating program be customized for a particular system?
- If the operating program is very large, it may take a long time to create the digest of the whole flash memory on boot? Is there a method to speed startup to an acceptable time?
- What happens if the validation secret leaks out somehow? Can it be updated along with the signatures?
- Asymmetric algorithms like RSA and ECC don’t require the storage of a secret in the system, just a public key. Why not just use these and save the cost of the security device?
- How can the AES accelerator found in many standard microprocessors be used to generate the program digest quickly?
- How can the secret be programmed into the security device without divulging it to third party subcontractors?
- Are there other system security needs that can be addressed with the same ATSHA204 device?

See the extended applications note, “Implementing Secure Boot with the Atmel ATSHA204,” for a more detailed description of the secure hash procedure along with answers to all these questions.

1. Revision History

Doc. Rev.	Date	Comments
8788A	03/2012	Initial document release

Enabling Unlimited Possibilities®

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan G.K.
16F Shin-Osaki Kangyo Bldg.
1-6-4 Osaki, Shinagawa-ku
Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370