Q&A for Tech Talk Topic: Device & Network Security for the IoT (April 21st, 2020)

Q: What is the best way to get a privat key/secret from the device owning into a Secure Element? At factory floor with a HSM?

A: There are several ways to provision secrets securely in an untrusted production environment.
A universal solution is to have a trusted entity (such as Silicon Labs, if you trust us) provision a bootloader image with Secure Upgrades enabled, public key, and decryption key into the device and lock the debug interface. At the point your CM receives the devices, they will only be able to load code bundles that you encrypt and sign into the device, and any other images will be rejected by the device. Vault products also have a means of establishing a secure provisioning channel between the SE CPU and the test infrastructure, so any information sent over that interface is held confidential.

The following documents might also be interesting for you:

"AN1218: Series 2 Secure Boot with RTSL"
https://www.silabs.com/documents/public/application-notes/an1218-secure-boot-with-rtsl.pdf

"AN1222: Production Programming of Series 2 Devices"
https://www.silabs.com/documents/public/application-notes/an1222-efr32xg2x-production-programming.pdf

"UG162: Simplicity Commander Reference Guide"
https://www.silabs.com/documents/public/user-guides/ug162-simplicity-commander-reference-guide.pdf

"UG266: Silicon Labs Gecko Bootloader User's Guide" describing the secure bootloader features
https://www.silabs.com/documents/public/user-guides/ug266-gecko-bootloader-user-guide.pdf

Q: Does Silabs provide to its customers Data generation service? Does Silabs have HSM to do it?

A: Because we have HSMs in our production infrastructure, we have the ability to provision devices with customized device certificates that are signed into a Silicon Labs certificate chain. Other types of customization can be supported as well. We can share more details if you reach out to your local Silicon labs sales office.

Q: You spoke about the root secret of all manufactures chips to be part of Silicon Labs CA chain. Can a customer also use their own root CA and cert chain to sign the devices at chip factory?

A: We do support various certificate customization options, including provisioning devices that are issued via a third-party CA. For more information, reach out to your local Silicon Labs Sales office for more info.

Q: With all these protection, it's a big chance that the SW developer get locked out of the chip. If and when this happenes, is a replacement of the chip the solution or is a total erase enough?

A: In most cases, developers will be using unlocked devices during development, so the risk of inconvenience caused by debug lock during the development process is low. If you lock the device using Standard Debug Unlock, the debug interface will be unlocked and as a side effect will erase the Flash memory and the RAM. If you lock the device using Secure Debug Lock, unlocking the device requires a signed cryptographic challenge token, which means you need to have access to the correct private key in order to generate that token. If you lose that private key or if you mis-program the Public key that gets loaded into the device, then the device will be locked forever.

Please also see AN1190 for more details:
https://www.silabs.com/documents/public/application-notes/an1190-efr32-secure-debug.pdf

Q: On secure key storage: why is storing a key in flash not secure in a series 0 device when the debug interfaceis disabled? Extraction would need decapping the chip? Or is that a false assumption?

A: You are correct. If the debug interface is locked (and no other interface is vulnerable), the attack is via decap + flash extraction. The problem is that flash extraction as a service is now relatively inexpensive. Reputable firms in Europe can charge $15k USD to extract up to 1 MB of flash on a 90nm process node. In other geographies the cost can be much less. Using a PUF-derived key, like we do in Vault secure key storage, doesn't prevent an invasive attack like this, but it makes getting the key much harder because the attack must be performed on a live device and it requires 256 probe points to be sampled, one for each bit in the PUF key.

Q: To use the anti-rollback protection for firmware updates: is there then a predefined way to code the version number in the firmware image? How does this in practice work?

A: You're right that the version number cannot be complex (like x.y.z.a), but must be monotonically increasing. The version number is a 32-bit unsigned value. If you enable anti-rollback and then load a version 0xFFFFFFFF of the code, you will effectively disable any further code updates.

Q: Interesting talk. Somewhat side angle question. I support the drive to security in IoT. Yet, there is an equally large drive from America and Europe not to export hardware encryption. This creates a large business risk to companies external to America and Europe (South Africa) Can you comment on this?

A: You are correct that trade restrictions can be confusing and can present barriers. I can't comment regarding other export rules, but in the US, if the cryptography is used as part of the product's function and is not end-user configurable, in general it is not subject to export restrictions. I am not well-versed in trade law, so you will need to get advice from a specialist who is familiar with your specific trade questions.