INTRINSIC ID

WHITE PAPER

Democratizing Roots of Trust from Silicon to Software

With a vast amount of devices getting connected to the internet of things (IoT) and the growing number of low-cost attacks being developed to hack such IoT devices, it is clear that the need for embedded security solutions is rising dramatically. A security subsystem in the main system-on-chip (SoC) of a device can be deployed to offer secure cryptographic services to the applications running on the device. But these services can only be secure if the cryptographic keys used by the system are protected properly against attackers. In practice, this is quite a challenge. Existing key storage methods offer limited security against invasive attacks, are inflexible and can add significantly to costs. A universal solution for solving this key storage problem is offered by SRAM physical unclonable function (PUF) technology. In this whitepaper we explain the basics of this technology and show how an embedded software solution is able to leverage PUF technology to add a strong, secure root key to almost any IoT device, without requiring changes in the SoC's hardware.

SRAM PUF Benefits

- Uses standard SRAM
- Device-unique, unclonable key
- No secrets reside on the IC
- No key material programmed
- Supply chain simplicity
- Low cost

Certifications

- EMVCo, Visa
- CC EAL6+
- NIST CAVP
- PSA, ioXt
- U.S. and EU Governments

Markets

- IoT
- Data Center / HPC
- Secure Transactions
- Aerospace & Defense

Greater Security for All IoT Devices

It is estimated that by 2025, there will be around 80 billion devices connected to the internet[1]. By interconnecting billions of devices in the internet of things, the world has become exposed to a plethora of security-related threats that never existed before. While companies struggle to recover from the damage caused by today's cyber-attack, attackers are fabricating new low-cost attacks using increasingly cheaper tools to attack IoT devices. The need for more security is clear. A recent publication from the Global Semiconductor Alliance (GSA)[2] outlines the concept of a security subsystem, integrated into a larger microcontroller or system-on-chip (SoC) controller, which in turn is at the heart of an IoT device. As part of a larger system, this security subsystem would provide services to applications, including managing and protecting digital assets such as certificates and cryptographic keys. These assets form the basis of any security architecture and are therefore essential. Cryptographic keys have many purposes, such as verifying a device's identity, securing communication between devices, and encrypting sensitive data at rest as well as in transition. Some of these assets, such as a decryption key or a signing key, are very sensitive and need to be protected and stored in a highly secure way.

Whether using a security subsystem, or some other method, there is a clear need for greater security for all IoT devices. Figure 1, illustrating a low-cost IoT device transmitting sensor data to a cloud service, provides an overview of the different security challenges that need to be addressed. This paper describes the existing methods for getting critical cryptographic assets into devices and for keeping them secure. In addition to a review of the traditional methods, we take a detailed look at the use of SRAM Physical Unclonable Functions or PUFs for this purpose. In particular, the software implementation of an SRAM PUF system has the potential to address the IoT security needs at a lower cost and with more ease and flexibility, all of which are important to IoT device makers.

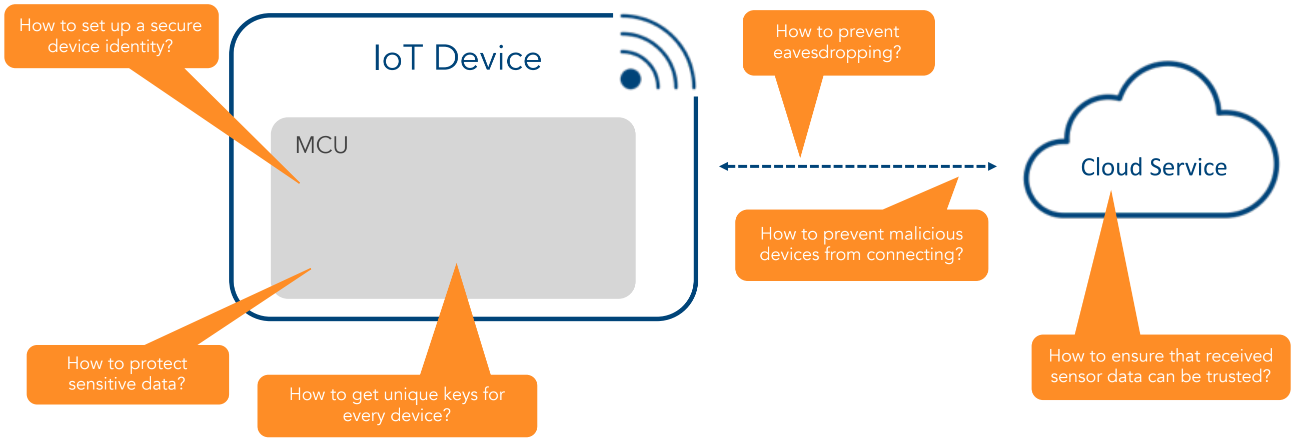

How to set up a secure device identity?
IoT Device
MCU
How to prevent eavesdropping?
Cloud Service
How to prevent malicious devices from connecting?
How to protect sensitive data?
How to get unique keys for every device?
How to ensure that received sensor data can be trusted?

Figure 1. Security challenges for an IoT device.

[1] https://www.forbes.com/sites/michaelkanellos/2016/03/03/152000-smart-devices-every-minute-in-2025-idc-outlines-the-future-of-smart-things/?sh=13391c2d4b63

[2] GSA white paper: "Preventing a $500 Attack Destroying your IoT Devices" https://www.intrinsic-id.com/resources/white-papers/landing-page-white-paper-preventing-a-500-attack/

Key Generation and Secure Storage

In this section we discuss the most widely used traditional methods for establishing identities and provisioning keys for IoT devices. For simplicity, we limit the discussion to the provisioning of a cryptographic root key that serves as the foundation for the device's security. This root key should never leave the security subsystem of the IoT device. When a device has a root key, all other cryptographic keys and identities of the device can be derived and/or protected from this root key, to create a chain of trust for the device as explained in "Flexible Key Provisioning with SRAM PUF."[3]

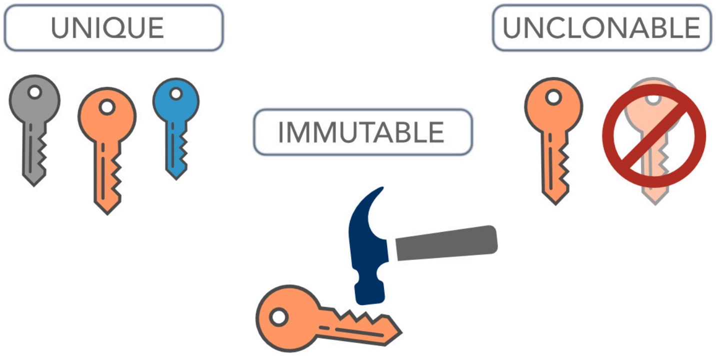


Figure 2: To establish trust in the IoT, devices need unique keys that are protected from attackers.

The root key is the main secret on which all device security is based, so the root key should be protected against:

- Readout by attackers: this would give an attacker the opportunity to decrypt communications and stored data and change trusted functionality, thereby compromising the entire system
- Altering by attackers: with altered keys, attackers would be able to install their own malicious code on legitimate devices
- Copying to other devices: this would allow attackers to create working clones of a device, which could lead to counterfeit devices on the black market or unauthorized devices in IoT networks

To ensure the device and its functions – the combination of hardware and software, its data, and its communications – can be protected, a device's root key must be immutable and unreadable by (cyber) attackers.

Traditional Methods

There are several traditional methods for generating and storing keys (including root keys) and other confidential data on an IoT device. Below, we briefly review a few of the most popular approaches and their strengths and weaknesses.

Secure Elements

Globalplatform.org defines a secure element (SE) as a tamper-resistant platform (typically a one-chip secure microcontroller) capable of securely hosting applications and their confidential and cryptographic data (for example cryptographic keys) in accordance with

[3] Intrinsic ID white paper: https://www.intrinsic-id.com/resources/white-papers/protecting-iot-invisible-keys-white-paper/

the rules and security requirements set by well-identified trusted authorities.[4]

The use of a secure element is more costly and complex

Secure elements destined for IoT devices are typically purchased from a silicon vendor with all required keys pre-provisioned on the chip by this vendor. This means that the IoT device maker does not need to provision keys for their device, but the SE approach comes with significant downsides, such as increased costs and complexity in purchasing, supply chain and inter-chip interfacing.

Key Injection

Another option for storing keys on IoT devices is injecting keys into the chip. Using this approach, the root key is generated outside the electronic device and injected during the production process. Typically, this needs to take place at an early stage in the supply chain (e.g., at the chip maker), because many parties in the supply chain will need to make use of the root key, for example at chip distribution or device manufacturing.

Key injection typically takes place early in the supply chain

After injection, the root key is stored on the device. Most widely used embedded key storage methods are based on non-volatile memory (NVM) such as electrically erasable programmable read only memory (EEPROM), Flash, or one-time programmable (OTP) memories such as fuses and anti-fuses. With these memory types, the provisioning of root keys comes with trade-offs among flexibility, key-exposure liability, cost, reliability, and security.

Random Number Generation

The third method for provisioning keys for IoT devices is to use an internal random number generator (RNG) on the chip for generating a random secret, which is then stored in NVM. This means key generation is handled internally, but key storage remains the same as with key injection. Using this method increases the flexibility within the supply chain compared to key injection (assuming the target chip contains a random number generator), but the same kinds of trade-offs seen in key provisioning hold true for RNG-derived keys as well.

Using an RNG, key generation can be handled internally

The SRAM PUF Based Method

An alternative approach to these traditional methods of generating and storing root keys is an SRAM PUF. SRAM PUFs use the behavior of standard SRAM memory, available in any digital chip, to extract a unique pattern or "silicon fingerprint." This pattern is virtually impossible to clone or predict. This makes SRAM PUFs very suitable for extracting a secure device-unique root key within the security subsystem of a device.

PUFs are virtually impossible to clone or predict

The next sections explain how SRAM PUFs work and how they can be used to derive a reliable root key that is never stored nor injected. In this way, SRAM PUFs provide a highly secure way of managing a root key in the security subsystem inside an integrated circuit.

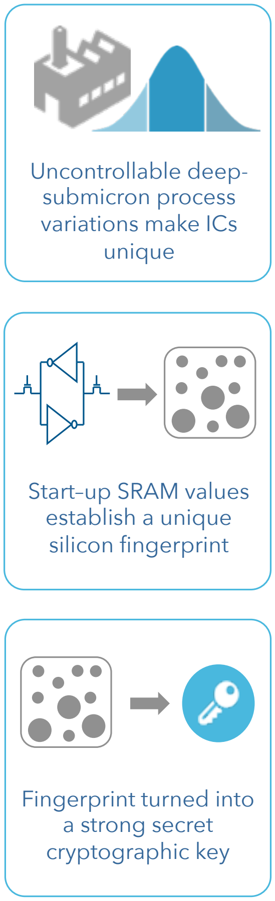
[4] https://globalplatform.org/resource-publication/introduction-to-secure-elements/

The SRAM PUF

Due to deep submicron manufacturing process variations, every transistor in an integrated circuit (IC) has slightly different physical properties. These lead to small but measurable differences in terms of electronic properties such as transistor threshold voltage and gain factor. Since these process variations are not fully controllable during manufacturing, these physical device properties cannot be copied or cloned.

Threshold voltages are susceptible to environmental conditions such as temperature so their values cannot be used directly as unique secret keys or identifiers.

The PUF behavior of an SRAM cell, on the other hand, depends on the difference of the threshold voltages of its transistors. Even the smallest differences will be amplified and push the SRAM cell into one of two stable states. Its PUF behavior is therefore much more stable than the underlying threshold voltages, making it the most straightforward and most stable way to use the threshold voltages to build an identifier.



Uncontrollable deep-submicron process variations make ICs unique

Start-up SRAM values establish a unique silicon fingerprint

Fingerprint turned into a strong secret cryptographic key

Figure 3: Extracting a strong secret key from SRAM behavior.

SRAM PUF Behavior

An SRAM memory consists of a number of SRAM cells. Each SRAM cell consists of two cross-coupled inverters that each are built up by a p- and n-MOS transistor. When power is applied to an SRAM cell, its logical state is determined by the relation between the threshold voltages of the p-MOS transistors in the invertors. The transistor that starts conducting first determines the outcome, a logical “0” or “1.”

It turns out that every SRAM cell has its own preferred state every time the SRAM is powered, resulting from the random differences in the threshold voltages. This preference is independent from the preference of the neighboring cells and independent of the location of the cell on the chip or on the wafer.

Hence an SRAM region yields a unique and random pattern of 0s and 1s. This pattern can be called an SRAM fingerprint since it is unique per SRAM and hence per chip. It can be used as a PUF.

Keys that are derived from the SRAM PUF are not stored “on the chip” but they are extracted “from the chip,” only when they are needed. In this way they are only present in the chip during a very short time window. When the SRAM is not powered there is no key present on the chip, which makes the PUF-based key-generation mechanism highly secure against invasive attacks.

Key Generation and Storage Based on SRAM PUF

Intrinsic ID provides the intellectual property (IP) that turns the slightly noisy fingerprint of the SRAM start-up response into a reliable root key. Whenever the root key is needed by the system, the IP reliably reconstructs it, eliminating the need for storing this root key in any form of memory. This means that when the device is powered off, no secret key can be found in any memory; in effect, the root key is “invisible” to hackers.

A whole tree of cryptographic keys (starting from the PUF-based root key) can be (re-)created without storing them in a memory, removing the need for a device to have any physical form of secure storage. More details about the basic functionality of SRAM PUF can be found in "SRAM PUF: The Secure Silicon Fingerprint,"[5] while details about how to use this technology for key provisioning can be found in "Flexible Key Provisioning with SRAM PUF."[6]

PUF Reliability

The deep submicron process variations that determine PUF behavior are created during manufacturing and do not change afterwards. Hence the start-up state preference of the SRAM cells is persistent and stable over time. However, there is still a degree of noise. A small number of the cells, whose start-up state is close to equilibrium, are unstable and display a seemingly random start-up preference. So, each time the SRAM starts up, a slightly different pattern emerges. This noise component is dependent on operating conditions such as temperature and voltage ramp-up.

The noise of SRAM-based PUF responses has been exhaustively characterized and tested under a wide variety of circumstances and foundry processes. The SRAM PUF has been qualified for automotive, industrial, and military use in collaboration with customers and partners. During these qualification processes, millions of measurements have been performed at varying conditions:

- Temperatures ranging from -55°C to +150°C [-67°F to 300°F]
- Voltage variation +/-20%
- Humidity up to 80%
- EMC tests at 3V/m (EN55020 0.15–150 MHz and IEC 61000-4-3 80-1000MHz)

Under all these circumstances the average noise level of the SRAM-based PUF response was found to be sufficiently low to be able to reconstruct a high-entropy, device-unique, and reliable key every time the SRAM is powered, by applying error-correction techniques such as 'helper-data algorithms'[7] or 'fuzzy extractors'[8]. These algorithms perform two main functions that will be explained below: error correction and privacy amplification.

Error Correction

Error correction techniques for cryptographic key reconstruction require an enrollment phase and a reconstruction phase. In the enrollment phase (a one-time process) the PUF response is mapped onto a codeword of an error correcting code. Information about the mapping is stored in an activation code (AC), sometimes called "helper data." The AC is constructed such that it does not provide any information about the key. It should be stored in memory that is accessible by the PUF algorithms but it can be stored off-chip as it is

[5] Intrinsic ID white paper: https://www.intrinsic-id.com/resources/white-papers/white-paper-sram-puf-secure-silicon-fingerprint/

[6] Intrinsic ID white paper: https://www.intrinsic-id.com/resources/white-papers/white-paper-flexible-key-provisioning-sram-puf/

[7] J.-P. Linnartz and P. Tuyls, "New shielding functions to enhance privacy and prevent misuse of biometric templates," in International Conference on Audio and Video-based Biometric Person Authentication (AVBPA'03), ser. LNCS, J. Kittler and M. S. Nixon, Eds., vol. 2688. Heidelberg: Springer-Verlag, 2003, pp. 393–402.

[8] X. Boyen, "Reusable cryptographic fuzzy extractors," in ACM Conference on Computer and Communications Security (CCS'04). New York, NY, USA: ACM, 2004, pp. 82–91. AND Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in EUROCRYPT'04, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Heidelberg: Springer-Verlag, 2004, pp. 523–540.

not sensitive. Any change to the AC, malicious or not, will prevent key reconstruction. Because the AC is created from a device-unique PUF response, the AC is only valid for the chip on which it was created.

Each time the device runs an authentication protocol and needs the secret PUF key, a new noisy PUF measurement is carried out and the PUF key (without noise) is extracted from the AC and this new PUF response. This is called the reconstruction phase. The error correction algorithms have been designed to reconstruct the key with a typical error rate[9] of less than 10^{-12}. Both enrollment and reconstruction phases are illustrated in Figure 4.

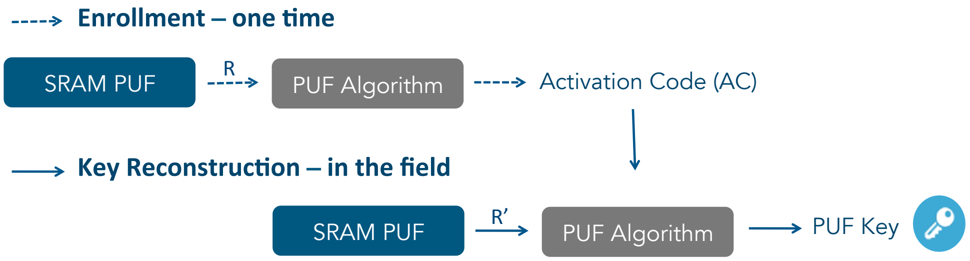


Figure 4: Enrollment and reconstruction phase for the generation of PUF keys. Note that R is the initial PUF response during enrollment while R' is a PUF response in the field with a noise component.

Privacy Amplification and Security

Secret keys provide security based on the fact that they are completely random and hence unpredictable. Physical measurements, such as PUF responses, have a high degree of randomness, but are usually not completely uniformly random. Privacy amplification is used to generate uniformly random keys.

By combining error correction and privacy amplification, a 1kByte SRAM PUF response can be turned into a 256-bit uniformly random key; only approximately 0.5 kByte is needed for a 128-bit key with full randomness.

Dedicated security labs and security teams at customers have analyzed the security of the Intrinsic ID SRAM PUF based IP solutions against various invasive and non-invasive physical attacks without revealing any weaknesses. Attacks with scanning electron microscopes (SEMs), lasers, focused ion beams (FIBs), and probes have not been successful. Side-channel attacks have not led to any leakage of sensitive information.

Aging

Accelerated aging tests have been performed on SRAM-PUFs to investigate the noise level as a function of time. By using a patented anti-aging technique[10] a 25-year lifetime can be guaranteed for SRAM PUF technology.

[9] Even under extreme circumstances e.g. due to extreme temperatures, if noise levels were to rise up to 25%, the reconstruction failure rate is still lower than 10^{-9}.

[10] R. Maes and V. van der Leest, "Countering the effects of silicon aging on SRAM PUFs", Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST), pp. 148-153 available at http://www.Intrinsic.id.com/wp-content/uploads/2014/09/PUF_aging.pdf

Device Maker Priorities	SRAM PUF	Internal RNG + key storage in NVM	Key Injection + key storage in NVM	Secure Element
Robust root key storage	✓	X Key stored in clear		✓
Outsource security to experts	✓	✓	✓	X Interfacing to SE still required
Low cost	✓	✓	X Service fee	X Extra chip required
Supply chain simplicity	✓	✓	X Keys need to be injected	X Extra chip/supplier required

Table 1: Comparison of the different methods for getting keys in devices with regard to device maker priorities.

Chip Supplier Priorities	SRAM PUF	Internal RNG + key storage in NVM	Key Injection + key storage in NVM	Secure Element
Integration of functionality	✓	✓	✓	X Integration external SE required
Differentiation on security	✓	X Any vendor can offer this service	X Any vendor can offer this service	X External SE required
No key provisioning service required	✓	✓	X	✓ (done by SE provider)
Process independent solution for key storage	✓	X	X	N/A External chip

Table 1: Comparison of the different methods for getting keys in devices with regard to chip supplier priorities.

Comparing the Different Key Storage Methodologies

In Tables 1 and 2 we compare and summarize how the different technologies for key storage discussed in this paper address the individual needs of device makers and chip suppliers, as well as their common need for secure key storage. Based on this comparison, it is clear that SRAM PUF offers the best combination of security, cost, integration and low complexity for generating root keys.

The only method whose performance comes close to that of SRAM PUF is use of an internal random number generator that stores a key in NVM, only because the keys are generated by similar methods. But the level of security robustness that is achieved for the actual storage of root keys is significantly lower for traditional key storage connected to an internal RNG compared to SRAM PUFs (see Figure 5). To explain this further, we take a closer look at the security properties of storing keys in NVM and with SRAM PUF.

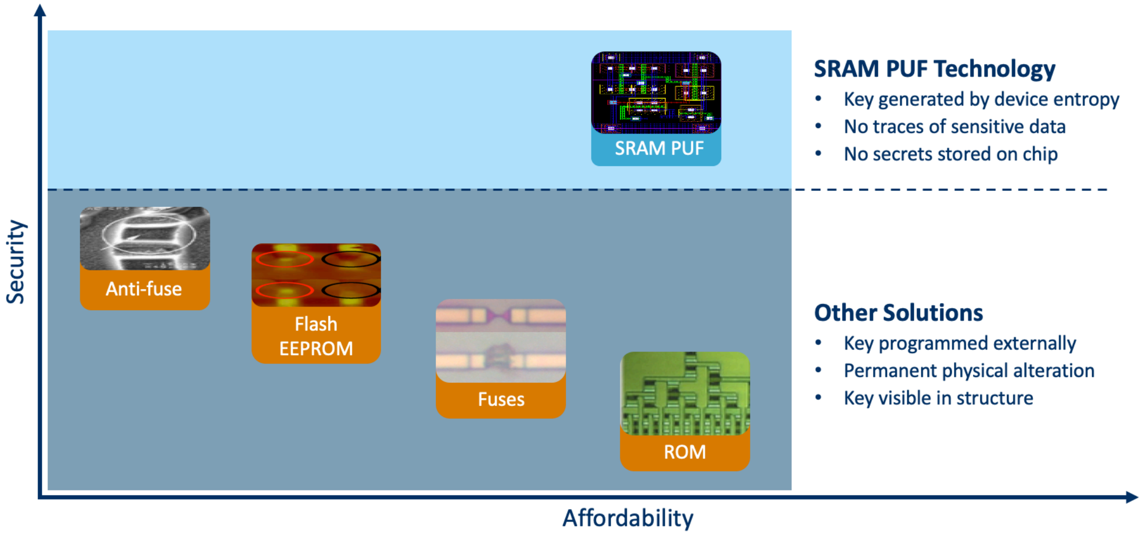

SRAM PUF Technology
Key generated by device entropy
No traces of sensitive data
No secrets stored on chip
SRAM PUF
Anti-fuse
Flash EEPROM
Fuses
ROM
Other Solutions
Key programmed externally
Permanent physical alteration
Key visible in structure
Security
Affordability

Figure 5: Security robustness versus affordability for the different key storage mechanisms.

Storing Keys in NVM

The number of companies offering legitimate reverse engineering services for hardware (for example, Tech Insights[11]) is growing. It is their business to help companies do studies on patent infringement and help chip suppliers expose weaknesses in their own security designs. However, the technology required to reverse engineer chips is also available to hackers and counterfeiters. And they are applying this on a wide scale to attack existing chips in the field.

One of the first and easiest steps in reverse engineering is opening up a chip and reading the content of NVM[12] such as read-only memory (ROM), Flash or EEPROM. Therefore, storing root keys in these memory types cannot be considered safe practice, as it has been shown on many occasions[12,13,14] that this sensitive material can be extracted with relative ease. To prevent sensitive data from being usable after it is extracted from Flash, for instance, the data should be encrypted before it is stored. For an attacker, extracting encrypted data is useless without the key. However, if only the type of storage available on the device is NVM, the key which has been used to encrypt the data must be stored in these easily compromised features. This is why chip suppliers generally look for other methodologies to store sensitive data on their chip.

OTP memory is a type of NVM which can be programmed only once, after which it cannot be changed. However, this is another memory type that permanently stores key material, leaving the possibility for attackers

[11] http://www.techinsights.com/

[12] University of Cambridge, Sergei Skorobogatov, April 2005: Semi-invasive attacks – A new approach to hardware security analysis, https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-630.pdf

[13] University of Cambridge, Sergei Skorobogatov, 2010: Flash Memory 'Bumping' Attacks. https://www.cl.cam.ac.uk/~sps32/ches2010-bumping.pdf

[14] Ohio State University and University of Michigan, multiple authors, 2018: NVCool: When Non-Volatile Caches Meet Cold Boot Attacks. https://xiangpan-osu.github.io/nvcool_iccd18.pdf

to find the physical residue that comprises the value of the stored secret. From a security point of view, anti-fuses generally are considered the best NVM for storing secret material. An anti-fuse is much more difficult to read out with an optical attack than other forms of OTP, such as regular fuses.[15]

However, even for these more complex OTP memories, attacks that read out the secret value successfully are on the rise[16]. An additional downside of anti-fuse technology is that it is process-specific and not always part of the standard CMOS manufacturing process, which creates additional overhead and costs in chip manufacturing.

Storing Keys With SRAM PUF

As described previously, when using SRAM PUFs, cryptographic keys and identities are derived from a digital fingerprint in the start-up behavior of SRAM cells. This means the secret material is never stored in memory and no physical traces can be found on a chip that could lead to the discovery of secret material. Hence, using SRAM PUFs protects secrets from reverse engineering attacks, simply by virtue of the fact the secrets are not present on the chip in any physical form. The SRAM PUF not only removes the requirement for externally provisioning keys to the chip (because they are created from the inherent silicon imperfections), but also provides a level of security that cannot be achieved with any other form of key storage, since keys are not physically stored on the chip. This provides devices with unclonable, immutable, and essentially invisible keys.

This technology has been silicon-proven, having been used to secure more than 350 million devices, ranging from high-end security systems-on-chips to low-cost microcontrollers. Deployment of SRAM PUFs for key generation and storage in the IoT market is increasing rapidly and provides devices with an unclonable, immutable, and essentially invisible unique identity.

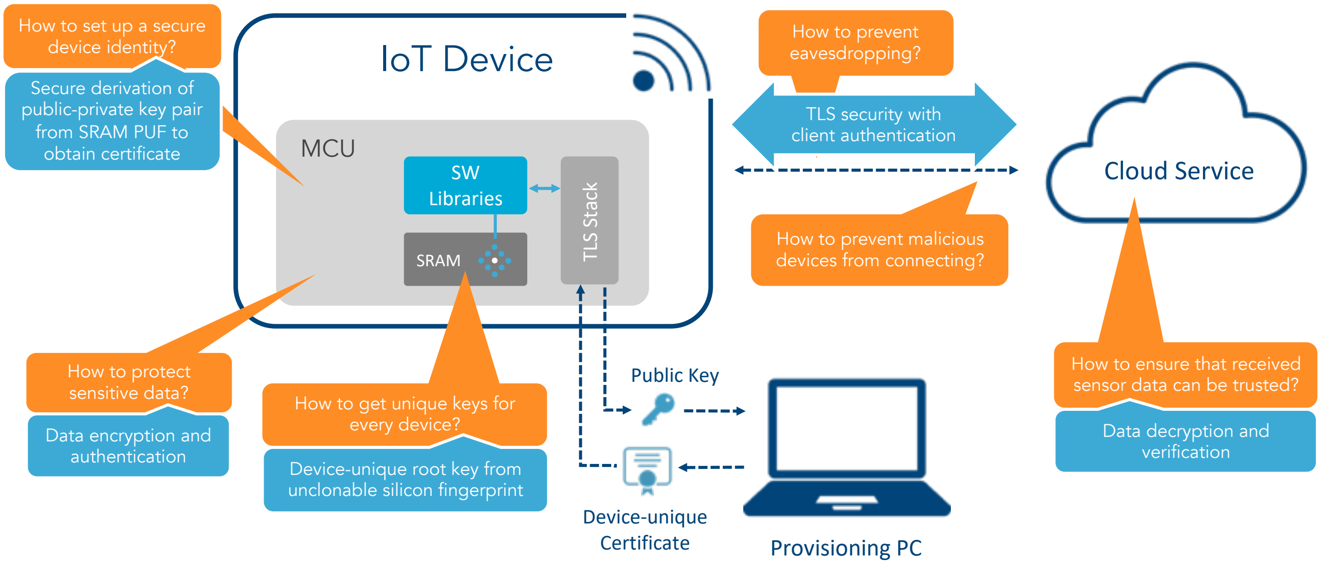

How to set up a secure device identity?
Secure derivation of public-private key pair from SRAM PUF to obtain certificate
IoT Device
MCU
SW Libraries
SRAM
TLS Stack
How to prevent eavesdropping?
TLS security with client authentication
How to prevent malicious devices from connecting?
Cloud Service
How to protect sensitive data?
Data encryption and authentication
How to get unique keys for every device?
Device-unique root key from unclonable silicon fingerprint
Public Key
Device-unique Certificate
Provisioning PC
How to ensure that received sensor data can be trusted?
Data decryption and verification

Figure 6: Chip-to-cloud authentication based on SRAM PUF.

[15] Chip Estimate, September 29 2015: Low-Power Embedded Memory Provides Superior Protection for IoT Devices, https://www.chipestimate.com/Low-Power-Embedded-Memory-Provides-Superior-Protection-for-IoT-Devices/Kilopass-Technology-a-part-of-Synopsys/Technical-Article/2015/09/29/

[16] Military Embedded Systems, blog: http://mil-embedded.com/articles/ensuring-versus-oxide-rupture/

Now that we have shown how SRAM PUFs provide a strong anchor for the root key, we can revert our attention back to the IoT example from Figure 1. This figure shows an IoT device connected to the internet and communicating with a cloud service. To do this in a secure way, for example via transport layer security (TLS), keys and identities are required. The trust anchor provides the means to generate and store the Public Key Infrastructure (PKI) asymmetric key pairs that are needed to establish the secure connections to the cloud, as well as provide the identity to uniquely authenticate the device when communicating. Figure 6 shows how SRAM PUF and cryptographic mechanisms around it remedy the challenges from Figure 1.

SRAM PUF Implementation With Ultimate Flexibility

Intrinsic ID provides embedded security IP solutions based on SRAM PUFs. These solutions include the mechanisms for error correction, randomness extraction and anti-aging as discussed above. Furthermore, advanced security countermeasures against fault injection and side-channel attacks are integrated into Intrinsic ID products. These products include a hardware IP (RTL netlist) implementation called QuiddiKey®[17] and a pure embedded-software-based product called BK™.[18]

A software implementation, such as BK, might be of interest in particular to IoT device makers since it can be used to add SRAM PUF technology to existing products through a simple firmware change. No dedicated hardware is required, as almost any digital chip already has the required SRAM memory available that can be used as PUF. So how does it work?

A Software Solution for Hardware-based Security

The BK software IP is provided as a compiled library for a specific CPU. The software integrator adds this library as part of its security software and reserves part of the chip's available SRAM memory for dedicated use by BK (by adapting the linker script according to the instructions in the BK manual).

It is important to note that, in contrast to other hardware-based security solutions, BK does not need to be loaded at silicon fabrication time, but can be installed later in the supply chain, and in extremis, even remotely retrofitted on deployed devices. This makes BK **the only software solution available today that creates a hardware-based security anchor**.

BK additionally provides cryptographic functionality* that is required for a robust security solution for any given IoT device, such as:

- Key derivation functions for extracting numerous keys from the internal PUF root key for various use cases within the device's firmware that require unique cryptographic keys
- Key wrapping functionality to support secure storage of sensitive data and keys that are generated on the device
- Random number generator functionality, seeded from the inherent noise in the SRAM power-up values, to create strong random numbers for supporting cryptographic protocols
- Cryptographic functions for generating and verifying digital signatures

[17] Intrinsic ID website: https://www.intrinsic-id.com/products/quiddikey/

[18] Intrinsic ID website: https://www.intrinsic-id.com/products/bk-software/

BK cryptographic functions use algorithms and schemes that are compliant to the NIST standards. BK comes in several configurations in terms of functionality (and using different amounts of resources). Using BK enables chipmakers to create a secure product with added value for IoT device makers, while keeping their process flexible and their costs low.

By choosing the appropriate configuration of BK, this solution can be applied to many different use cases, such as:

- Creating a device-unique cryptographic root key inside a security subsystem: Even the smallest instantiation of BK provides an SRAM PUF that creates a unique identity and a cryptographic root key for a device, which makes it very suitable for use in even the most resource-constrained MCUs.
- Secure vault: In a secure vault, any stored data is securely and physically bound to the hardware of the device. All sensitive data is encrypted and authenticated with keys derived from the PUF root key. The data can only be verified and unwrapped on the specific device upon which it was previously wrapped.
- Device authentication through asymmetric cryptography: Leveraging BK's asymmetric crypto functionality, a device unique certificate can be setup with help of a trusted certificate authority (CA). The device can prove its authenticity to other devices or servers in the network by proving ownership of the corresponding private key using BK's digital signature functionality.

Conclusion

The Intrinsic ID BK software solution provides an elegant way of protecting the root key of any IoT device. It provides a strong foundation for the device's security by leveraging the unique power-up properties of SRAM memory to extract a device-unique cryptographic root key that is never stored. Instead, it is only reconstructed on the fly when needed by the device's security subsystem. This has a clear security advantage over traditional key storage methods such as embedded flash and OTP memory, where a programmed key leads to physical state changes that are discoverable by attackers. Besides offering a security advantage, it also saves costs and adds flexibility since the root key does not have to be injected during manufacturing. The cryptographic root key never leaves the security subsystem and is used to derive a next layer of keys for supporting various cryptographic applications such as encryption and digital signatures. The underlying SRAM PUF technology has been proven in the field with more than 350 million deployments to date.

**The full functionality of BK can be explored by downloading and installing BK-Demo on a development board. BK-Demo is a simulation version mimicking the behavior and functionality of the commercial product BK and is available for free here: https://www.intrinsic-id.com/resources/bk-demo-cortex-m/.*

info@intrinsic-id.com   www.intrinsic-id.com   INTRINSIC ID

Intrinsic ID Inc., 710 Lakeway Drive, Sunnyvale, CA 94085 U.S.
Intrinsic ID B.V., High Tech Campus 83, 5656 AG Eindhoven, The Netherlands