INTRINSIC ID

# WHITE PAPER

## Protecting the IoT with Invisible Keys

### 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

## Rooting Trust in Silicon

With more than 10 billion devices connected to the Internet of Things (IoT), the need for strong security solutions cannot be understated. Security in IoT, or in any given situation, starts with trust. This white paper addresses IoT security by looking at the most fundamental assets of a connected device that need to be established: the keys and identities that that are essential to protect data and authenticate devices to the network and each other. These assets are at the heart of any security architecture especially for IoT where it is estimated that by 2025, there will be 152,200 devices connecting to the internet per minute. In this white paper we look at the challenges when generating and protecting secret keys such as the root key. In addition to traditional methods, a method based on static random-access memory (SRAM) physical unclonable functions (PUFs) is examined. We will show how trust in the IoT can be established with invisible keys rooted in silicon..

## Limiting the IoT Attack Surface

Studies of the attack surface of the IoT have shown that a majority of the threats and attack vectors stem from insufficient establishment of foundational trust in the IoT network.[1,2] The foundation of trust in the IoT is based on device identities and cryptographic keys. Unique device identities[3] are essential to identify each individual device, while cryptographic keys have many purposes, such as verifying the device's identity, securing the 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 securely.

## Key Generation and Secure Storage

To establish a foundation of trust, chip designers need to establish device identities and provision keys into their devices and keep these assets secure. In this section we discuss the most widely used traditional methods for establishing identifies and provisioning keys for IoT devices. For simplicity, we limit the discussion to the provisioning of the root key that serves as the foundation for all security on a device. When a device has a root key, all other cryptographic keys and identities of the device can be derived from this root key, to create a chain of trust for the device as explained in "Flexible Key Provisioning with SRAM PUF".[4]

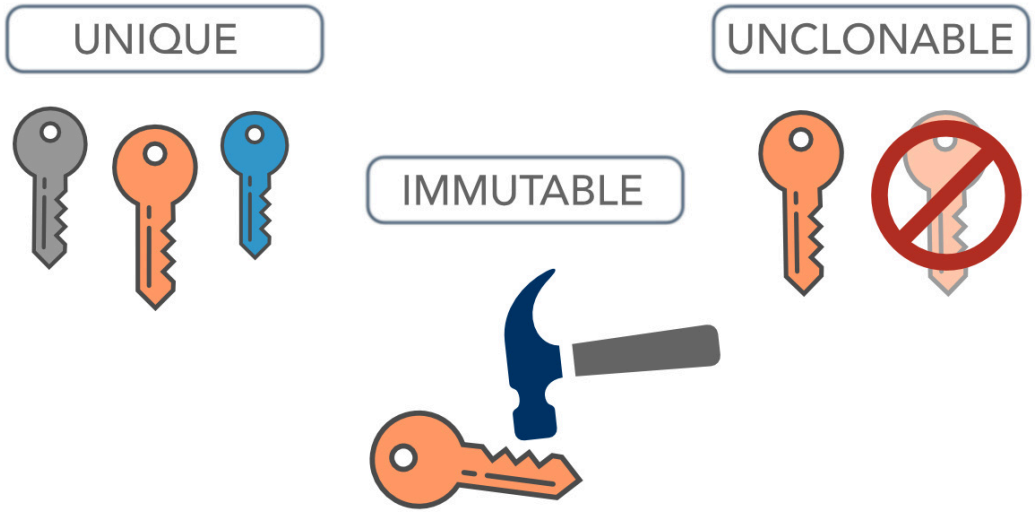



Figure 1: To establish trust in the IoT, devices need unique keys that are protected from attackers.

[1] Infosec Institute, February 17 2018: The Top 10 IoT Vulnerabilities, https://resources.infosecinstitute.com/the-top-ten-iot-vulnerabilities

[2] IBM Developer Blog, November 17 2017: Top 10 IoT Security Challenges, https://developer.ibm.com/dwblog/2017/iot-security-challenges/

[3] In this work we define device identity as follows: an identity comprises a set of cryptographic keys that are linked to a public identity value (e.g. a public/private key pair and a certificate that links the public key to a public identity value / UID / serial number).

[4] Intrinsic ID white paper: http://go.intrinsic-id.com/flexible-key-provisioning-sram-puf-lp

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

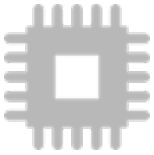
- Readout by attackers: this would give an attacker the opportunity to decrypt communications and stored data and change trusted functionality, thereby compromising the entire system
- Altering by attackers: with altered keys, attackers would be able to install their own malicious code on legitimate 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



*The use of a secure element is more costly and complex*

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

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

*Key injection typically takes place early in the supply chain*

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.

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.

*Using an RNG, key generation can be handled internally*

## Random Number Generation

The third method for provisioning keys for IoT devices is to use an internal random number generator (RNG) on the chip that requires a root key, which derives a random secret and stores it 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.

## The Alternative: Securely Generating Keys with SRAM PUFs

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." They are virtually impossible to clone or predict. This makes them very suitable for applications such as secure key generation and storage, device authentication, flexible key provisioning and chip asset management.

Due to inherent deep submicron process variations in the production process, every transistor in SRAM cells has slightly random electric properties. This randomness is expressed in the start-up values of uninitialized SRAM memory. These values form a unique chip fingerprint, called the SRAM PUF response. Even though these values are slightly noisy between start-ups of the same device, they can be turned into stable keys and form an excellent basis for the trust for a system.

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]

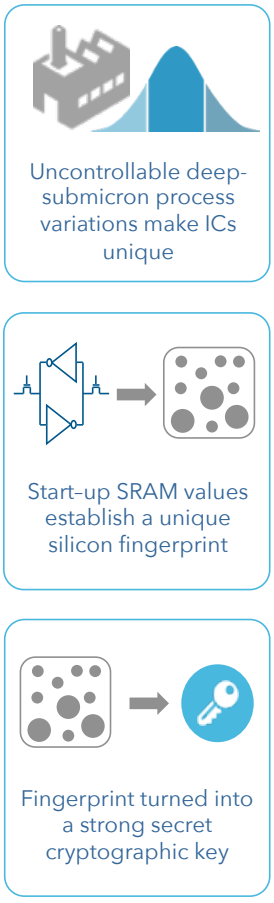



**Figure 2: Extracting a strong secret key from SRAM behavior.**

[5] Intrinsic ID white paper: http://go.intrinsic-id.com/secure-silicon-fingerprint-lp

[6] Intrinsic ID white paper: http://go.intrinsic-id.com/flexible-key-provisioning-sram-puf-lp

## Comparing the Different Methodologies

Table 1 offers a comparison between the three traditional methods to generate and store the cryptographic root key of a device and the SRAM PUF. Based on this comparison, the SRAM PUF offers the best combination of security, cost, and low complexity for generating root keys. The internal RNG is close to the SRAM PUF in terms of low cost and simplicity of supply chain, but any solution that stores a key in NVM will have less-robust security for stored root keys (and any other stored keys or sensitive data).

| | SRAM PUF | Internal RNG + key storage in NVM | Key Injection + key storage in NVM | Secure Element |
|---|---|---|---|---|
| **High Security** | ✓ | X Key stored in clear | | ✓ |
| **Supply Chain Simplicity** | ✓ | ✓ | X Injection needed | X Additional chip |
| **Low Cost** | ✓ | ✓ | X Service fee | X Additional chip |

Table 1: Physical unclonable function compared to other key generation and storage mechanisms.

## Storing Keys in NVM

To explain this further, let's take a closer look at the security provided by storing keys in NVM and by using an SRAM PUF (Figure 3).

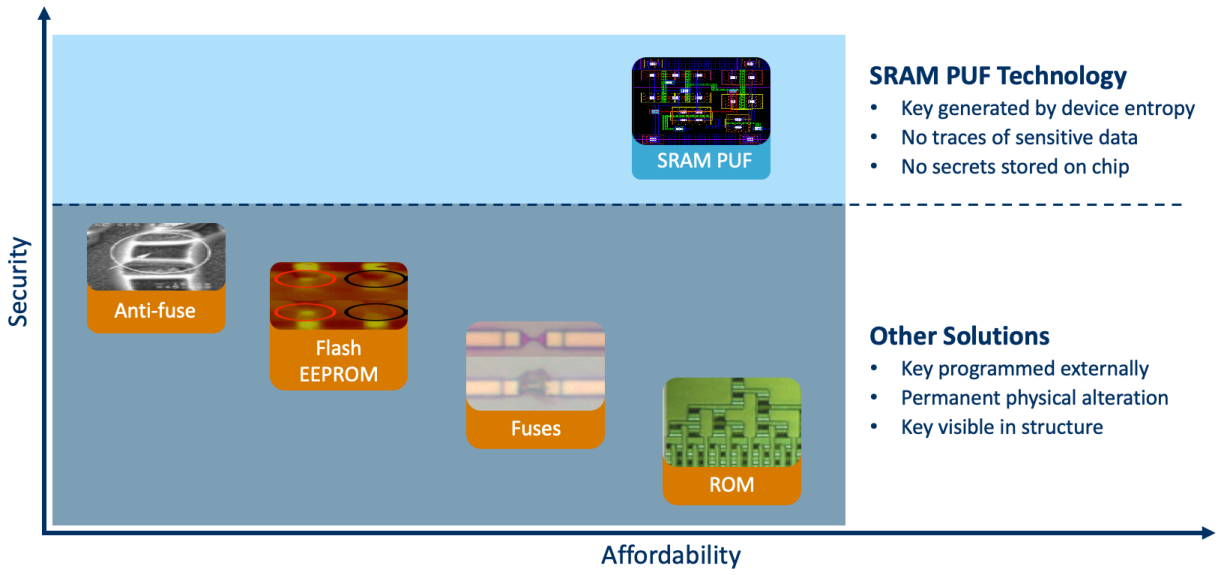



Figure 3: Security robustness versus affordability for the different key storage mechanisms.

The number of companies offering legitimate reverse engineering services for hardware (for example,Tech Insights[7]) 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 on the market.

One of the first and easiest steps in reverse engineering is simply to open up a chip and read the contents of NVM[8] 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 many occasions[9,10] 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, extracted 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 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.[11]

However, even for these more complex OTP memories, attacks that read out the secret value successfully are on the rise.[12] And even if the anti-fuse itself would be more resistant to optical reverse engineering techniques, the interface between the microcontroller and the anti-fuse memory is vulnerable and can be reverse engineered.[13] An additional downside of anti-fuse technology is that it is process-specific and is not always part of the standard CMOS manufacturing process, which creates additional overhead and costs in chip manufacturing.

---

[7] Website: http://www.techinsights.com/

[8] 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

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

[10] 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

[11] 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

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

[13] Verisiti white paper: http://verisiti.com/media/1025/ap4623_how-safe-is-anti-fuse-article.pdf

## Storing Keys with SRAM PUFs

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.

# Conclusion

SRAM PUFs offer the best combination of security, cost, flexibility and integration to establish a foundation of trust in the IoT. The technology combines high security and reliability with low-cost, low-footprint designs, and easy implementation, outperforming traditional methods for key generation and storage. For more than a decade, SRAM PUFs have been successfully implemented in commercial products, and have been deployed in hundreds of millions of devices, from tiny microcontrollers and sensors to high-performance FPGAs and secure elements.

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