Secure and Efficient Initialization and Authentication Protocols for SHIELD

Chenglu Jin and Marten van Dijk

University of Connecticut
chenglu.jin@uconn.edu, vandijk@engr.uconn.edu

June 27, 2015

Abstract

With the globalization of semiconductor production, out-sourcing IC fabrication has become a trend in various aspects. This, however, introduces serious threats from the entire untrusted supply chain. To combat these threats, DARPA (Defense Advanced Research Projects Agency) proposed the SHIELD (Supply Chain Hardware Integrity for Electronics Defense) program to design a secure hardware root-of-trust, called dielet, to be inserted into the host package of legitimately produced ICs. Dielets are RF powered and communicate with the outside world through their RF antennas. They have sensors which allow them to passively (without the need for power) record malicious events which can later be read out during an authentication protocol between the dielet and server with a smartphone as intermediary.

We propose the first concrete protocol design for initialization in SHIELD and an improved protocol design for authentication in SHIELD (compared to DARPA's call for proposals for SHIELD). As the basis for authentication we propose to use AES counter mode encryption (as opposed to DARPA's plain AES encryption). We show that this leads to several advantages: (1) resistance to a "try-and-check" attack which in case of DARPA's authentication protocol nullifies the effectiveness of one of SHIELD's main goals (that of being able to detect and trace adversarial activities with significant probability), (2) immunity against differential power analysis and differential fault analysis for free, (3) a $2\times$ speed up of the authentication phase by halving the number of communication rounds with the server, and (4) a significant reduction of the power consumption of the dielet by halving the number of needed AES encryptions and by halving the number of transmitted bits.

For initialization (each dielet needs to go through an initialization phase during which the manufacturer sets a serial ID and cryptographic key) we propose the first efficient and secure protocol where dielets generate their own serial ID and key by using a true random number generator (TRNG). The advantage of the proposed initialization protocol is that (1) dielets are able to efficiently generate their serial IDs and keys in parallel on the wafer during a trusted manufacturing process, (2) dielets communicate their key and serial ID to a trusted authentication server after insertion into host chips during a trusted assembly process (this avoids uploading keys and serial IDs to authentication servers becoming a bottleneck), and (3) transits between trusted fabrication and trusted assembly facilities do not need to be trusted (due to a one-time initialization mode construct).

The area overhead of our authentication and initialization protocols together is only 64-bit NVM, one 8-bit counter and a TRNG based on a single SRAM-cell together with corresponding control logic.

Contents

1 Introduction

Outsourcing IC (Integrated Circuit) fabrication has become mainstream in IC design, fabrication, testing and packaging. Even though outsourcing to a trustworthy manufacturing facility for legit-

imate IC fabrication and assembly is assumed by default, legitimately produced ICs still need to pass through the remainder of a supply chain which is not in one's own control and can therefore not be trusted. This opens a whole new range of serious threats including compromise of IP (Intellectual Property) Privacy, IC Overbuilding, Reverse Engineering, and Counterfeit ICs. Due to these attacks, semiconductor industry not only loses 4 billion dollars annually [1] but also untrusted (expired or malicious) hardware has become common in embedded systems.

In order to have a sound basis for trustworthy embedded systems one would ideally want to have an additional point of trust within the supply chain such that somehow tamper-evidence can be added to legitimately manufactured ICs (and supply chain attacks can be prevented or detected). A first approach is to be completely in control of IC packaging so that ICs will be equipped with trusted packages which are smart in that they are into some extent tamper-evident. In other words, if tampering happens, then the package will gather irreversible evidence of the tampering. This evidence can be seen or "read-out" later to allow verification of the authenticity and integrity of the IC inside the package.

DARPA (Defense Advanced Research Projects Agency) takes this approach to a new level: Their SHIELD (Supply Chain Hardware Integrity for Electronics Defense) program proposes to embed/insert an ineradicable hardware root-of-trust, called dielet, into the host package of every legitimately produced IC. The dielet is intelligent in that it is able to passively sense and record malicious behavior (such as unexpected exposure to light, vibration, etc.) and can be read-out at a later moment to gather any recorded tampering evidence. SHIELD goes beyond the simple first approach described above: Not the IC packaging process itself needs to be designed to offer tamper-evidence, one only needs to make sure that a process of inserting dielets into host packages of legitimately produced ICs is in place. This process of dielet insertion must be part of the trusted IC assembly since otherwise any (malicious) IC can be linked/bound to a valid dielet and later on pass identification and authentication as if the IC can be trusted to be what it claims to be through the dielet.

Notice that the proposed dielet technology transforms any host package into a tamper-evident one. Besides the smaller Trusted Computing Base (TCB) in the form of trusting the dielets and trusting the bond (= host package) between ICs and dielets (rather than trusting a larger overall tamper-evident packaging), SHIELD also provides the main advantage of backward compatibility: SHIELD technology is a sort of labeling technology which applies to already existing IC manufacturing and corresponding supply chains.

The main design features of a dielet are shown in Figure 1, the entire authentication system contains three parts: a dielet inserted in the package of the host chip, a smartphone and a secure remote server. The remote server stores the information for identification and authentication, such as serial ID and cryptographic key for each dielet. The communication between the server and the dielet requires an inexpensive appliance to read the dielet. A smartphone with a probe as a common appliance can be used in practice. The communication channel between the server and the smartphone is over a wireless network and over the Internet while the dielet connects to the smartphone via an RF (Radio Frequency) channel. Because the dielet is passively powered up through an RF transceiver module, the dielet has to be *both area and energy efficient*.

The dielet provides a unique permanent identification and implements sensors that are capable of measuring extreme temperature, light exposure, vibration, UV radiation, etc. As a consequence of passive power supply, the sensors must be passive (e.g., light exposure can be recorded using photo sensitive material) [2].

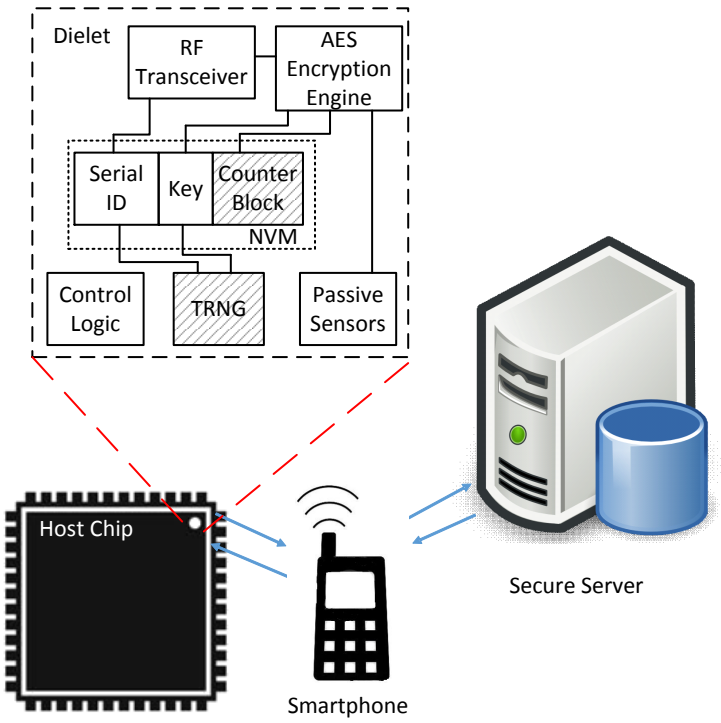

Figure 1: Authentication system of SHIELD. The shaded blocks are the modules that we propose to add.

The dielet has an encryption engine in order to encrypt the communication between the dielet and server (through the potentially untrusted smartphone). Some NVM (Non-volatile Memory) is embedded in the dielet to store a cryptographic key and a serial ID. We propose to add a hardware counter block (the top shaded block in Figure 1) in NVM in order to enable AES (Advanced Encryption Standard) [3] counter mode encryption [4]. This as we will explain will make the authentication protocol more secure, will speed up the authentication protocol, and will reduce the power consumption of the authentication protocol at the cost of one 8-bit counter in NVM as additional area overhead. In particular, we introduce the *"try-and-check" attack which in case of DARPA's authentication protocol nullifies the effectiveness of one of SHIELD's main goals (that of being able to detect and trace adversarial activities with significant probability)*. We show how our proposal prevents such attack.

Besides the authentication protocol we also need a protocol which initializes dielets with their own unique serial IDs and cryptographic keys. We propose to add a TRNG (True Random Number Generator) so that dielets can self-generate a serial ID and a secret key (the bottom shaded block in Figure 1).

1.1 Contributions

We introduce a new authentication and initialization protocol for SHIELD with the following advantages:

1.1.1 Performance Benefits

- We significantly reduce the dielet power consumption during authentication because of two crucial performance improvements:
  - Our proposed authentication protocol transmits only 208 bits between the dielet and smartphone, instead of 448 bits in DARPA's authentication protocol in the SHIELD call for proposals.
  - Our proposed authentication protocol uses only one AES encryption per authentication request as opposed to DARPA's protocol which needs two AES encryptions per request.
- Our design needs one 8-bit counter in NVM: We notice that the write latency of many emerging non-volatile memories (STT-RAM, PCRAM and ReRAM) vary from 10's ns to 100's ns, while NAND FLASH, as the most widely used NVM technology nowadays, has a write latency of $200\mu s$ [5]. Nevertheless, the write latency of FLASH is still negligible compared to the network communication latency between the smartphone and remote server, which are in the tens or hundreds milliseconds [6]. Therefore the speed of any authentication protocol is dominated by the relatively large network latency between the smartphone and server. For this reason our protocol achieves a $2\times$ speed up compared to DARPA's protocol by reducing two full communication rounds between the smartphone and server in DARPA's protocol to only one round in our protocol.
- In our proposed initialization protocol dielets generate their own serial ID and key by using a true random number generator (TRNG). This allows dielets to efficiently generate their serial IDs and keys in parallel on the wafer during a trusted manufacturing process. The resulting performance overhead is negligible. Also in our protocol dielets communicate their key and serial ID to a trusted authentication server after insertion into host chips during a trusted assembly process. Notice that it takes a significant amount of time (in the order of minutes) if keys and serial IDs are to be uploaded to an authentication server while the dielets are still on the wafer (since keys and serial IDs will need to be uploaded in sequence, i.e., one after another). Our approach avoids such a bottleneck.

1.1.2 Security Benefits

- Since DARPA's protocol uses plain AES which offers only *deterministic* symmetric key encryption, ciphertexts corresponding to the same plaintext in DARPA's protocol are linked over time. This makes their protocol particularly vulnerable to a simple *"try-and-check" attack which makes DARPA's protocol useless with respect to one of SHIELD's main aims*: The try-and-check attack nullifies an important role of the use of passive sensors in SHIELD in that adversaries are able to select which counterfeited/malicious chips have an assembled host package carrying a maliciously added dielet (taken from another legitimately produced host package) whose sensors did not detect any tampering (i.e., did not detect the removal and adding of the dielet from the legitimate host package to the maliciously assembled host package). So, adversaries are able to eliminate any trace or evidence of counterfeited/malicious chips that can be detected by the authentication protocol. Our proposal on the other hand prevents such attack, because it is based on AES in Counter Mode which is a probabilistic encryption scheme and therefore makes all ciphertexts look random to the adversary.

- The counter of AES in Counter Mode can be used as an extra sensor which teaches the authentication server how many times the dielet has been put into authentication mode when the dielet is offline with respect to the server. This can be used as an indicator of suspicious behavior.
- By setting a maximum to the allowed number of counter increments (which limits the dielet lifetime[1]), our protocol has immunity against (non-invasive) differential power analysis and differential fault analysis. Even if an attacker is willing to pay the price of extracting a serial ID and key pair, the pair can only be used up to the allowed maximum (otherwise the authentication server will detect that its synchronized counter exceeds the maximum). This demotivates an economically motivated adversary to extract serial ID and key pairs.
- Due to a one-time initialization mode construct in our initialization protocol, transits between trusted fabrication and trusted assembly facilities do not need to be trusted.

1.1.3 Area Utilization

The additional area utilization for our authentication+initialization protocols compared to DARPA's authentication protocol is only 4% of the allowed area of the dielet ($0.01mm^2$ [7]) in 32nm technology, which is very small compared to the area needed for AES, the passive sensors, RF transceiver, etc. using current state-of-the-art technology.

1.2 Organization

Section 2 provides background on supply chain security, discusses supply chain vulnerabilities and countermeasures, and introduces the trust model of SHIELD. Section 3 presents DARPA's suggested authentication protocol and describes our proposal of a secure and efficient alternative for SHIELD together with a performance and security analysis. In particular, we introduce the try-and-check attack. Section 4 explains our proposal of a new initialization protocol together with a performance and security analysis. Section 5 analyses the area overhead of our protocols by implementing the additional logic incurred by our protocols. Section 6 concludes the paper.

2 Supply Chain Security

2.1 IC Supply Chain Vulnerabilities

Supply Chain Security has recently gained significant interest in the hardware security community [8, 9]. [9] presents a detailed taxonomy of supply chain vulnerabilities, where supply chain vulnerabilities are classified into seven categories, as shown in Figure 2. (1) Cloned: Cloning is a common threat for IC design. Adversaries want to reduce the cost of design by cloning the design of others and produce their own chips illegally. This may happen during the design phase (copying the design files illegally) or distribution phase (reverse engineering the chip to obtain the design of a chip). (2) Tampered: Tampering with a chip is more commonly known as adding a Hardware Trojan, a research area that gained lots of interest in recent years [10]. Each phase of the supply

[1] We do not consider this to be an issue: In current supply chain management RFIDs that collect trace information are read-out by only a small/limited number of readers. Therefore, while in the supply chain until arriving at the end-user, we expect a dielet to be read out by a smartphone only a limited number of times.

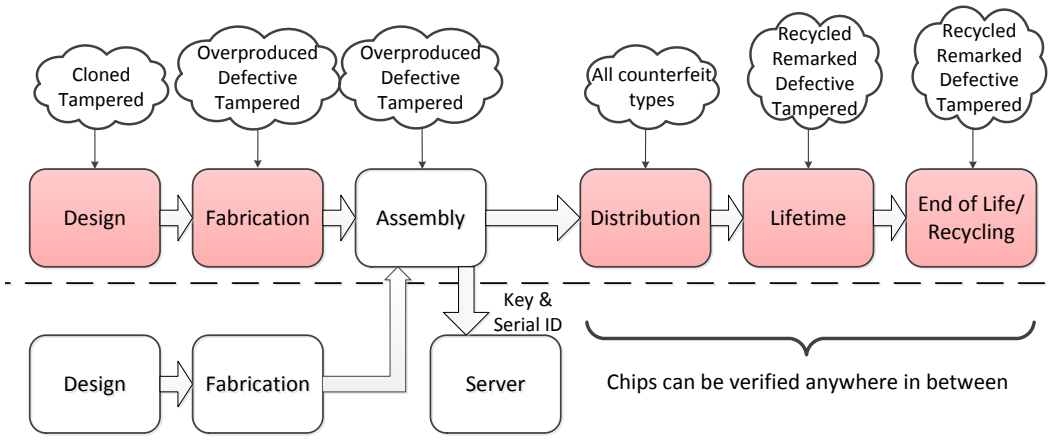

Figure 2: IC supply chain vulnerabilities and trust model for SHIELD. Dielet design, fabrication, insertion in IC host package, and initialization are each trusted. In addition either IC design, fabrication, and assembly must be trusted, or post-silicon Hardware Trojan detection and VLSI testing schemes must be employed.

chain is vulnerable to an adversary inserting a Hardware Trojan in order to initiate stealthy and malicious or destructive behaviors. (3) Overproduced: Since the globalization of semiconductor business, fabrication and assembly have largely shifted to foundry/assembly facilities, these facilities are capable of producing more chips than those promised by a contract or agreement. Untrusted facilities will illegally sell overproduced chips for extra profit. (4) Defective: Defective chips fail to correctly respond to at least one test vector. Yet, a defective chip may be sold as a functional chip on open markets by an irresponsible manufacturer who knows that the defect can only be excited in a corner case. (5) Recycled: After the lifetime of an electronic component, the component should be recycled properly. However, many recycled ICs are sold as new ICs after repackaging and remarking. (6) Remarked: In remarking, the adversaries remove the old marking on the chip and create a new coating for it with the aim to sell it with a higher specification. (7) Forged Documentation: Forged documentation includes a revision history of a component, certifications of compliance for some standards, etc.

2.2 Recent Detection/Avoidance Methods

Recently, researchers in academia and industry have proposed many counterfeit detection/ avoidance methods. Generally, these methods assign an ID to each chip and verify chips by checking these IDs. For example, silicon PUF (Physically Unclonable Function) technology [11, 12] can be used for IC identification and authentication since PUFs exploit the randomness of process variation during the fabrication phase to generate an unique fingerprint for each chip. It is impossible for an attacker to duplicate the PUF design with the exactly same process variation to regenerate the same chip ID. This is used to target remarked, overproduced, and cloned chips in the supply chain [9].

Another proposed countermeasure is Hardware Metering, which is a set of security protocols that enables design houses to achieve post-fabrication control of the fabricated IC. Usually hardware metering methods allow the designer to lock each chip through the fabrication, which means the

chips are not functional, and these locked chips can be unlocked only by the designers [13]. This prevents overproducing and cloning. In similar style, Secure Split Testing allows designers to protect and meter their design after fabrication by introducing the designs into a testing phase to prevent defective chips from entering the open market [14].

Split Manufacturing is another method to prevent untrusted foundries from overproducing chips. The designers split the layout of a chip into front-end-of-line and back-end-of-line layers, and fabricate these two layers in two different untrusted foundries (who are assumed not to collaborate). Next, the design house just needs a low-end trusted manufacturing facility to assemble these two layers [15].

Finally, lightweight aging sensors can be embedded into chips to detect recycled chips [16]. Since these sensors are capable of recording the usage time of a chip, the verifier can easily figure out for how long this chip has been used.

2.3 Trust Model for SHIELD

The trust model for SHIELD is illustrated in Figure 2. The supply chain above the dashed line is the original supply chain for IC design, and the one below is added by SHIELD to produce trustworthy dielets which are inserted in the host packages of legitimately produced ICs. The design, fabrication, insertion and initialization of dielets need to be trusted (white blocks in Figure 2).

Dielets allow the identity and authenticity of chips to be verified at any stages in the supply chain after the insertion process. Notice that even if the manufacturing facility cannot be trusted, overproduction can now be prevented by controlling and recording the number of used dielets. Concluding, SHIELD *alone without assuming a trustworthy manufacturing facility for ICs* has a near 100% coverage of all vulnerabilities in the entire supply chain, except for detection of possibly tampered and defective chips produced at the design and fabrication stages. Of course if SHIELD is combined with some post-silicon Hardware Trojan detection [17, 18] and VLSI (Very Large Scale Integrated circuit) testing schemes [19], all of the vulnerabilities in the whole supply chain are covered by SHIELD without the need for assuming a trustworthy manufacturing facility for IC design, fabrication, and assembly.

3 Authentication Protocol

In the call for proposals for SHIELD [7], DARPA suggests the authentication protocol as depicted in Figure 3. In order to verify the authenticity of the host chip, a smartphone with an inductive or RF probe plugged into it is used to first power up the dielet and upload the serial ID of the dielet to the server. Next, the server looks up the entry that corresponds to the serial ID in its database. If it is an existing serial ID in the database, the server generates a random challenge/nonce C and sends C to the dielet through the smartphone. The dielet encrypts the challenge and the sensor status bits SS using the on-board cryptographic key K, and sends the resulting ciphertexts X and Y back to the server. The server decrypts the ciphertext X using the key K' corresponding to the serial ID stored in the server's database, and compares the result C' with the original challenge C. If C' matches C and the sensor status bits do not record any attacks, then the server concludes that this chip is legitimately produced. In the last step the server replies the authentication result to the smartphone (to avoid man-in-the-middle attacks the smart phone and server are assumed to have their own authentic channel).

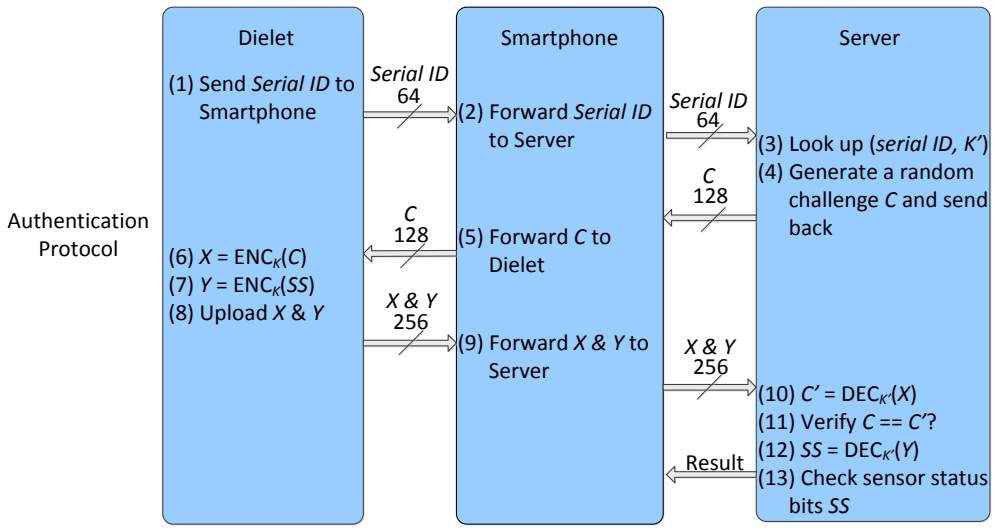

Figure 3: DARPA's Authentication Protocol. SS is sensor status. There are two full communication rounds between the smartphone and the remote server. It encrypts the challenge and sensor status bits separately. The total transmitted bits between the dielet and the smartphone are 448 bits.

3.1 Our Proposal

As explained in the introduction, we propose to base the authentication protocol on AES counter mode: Counter (CTR) mode [4] is a mode of operation for block ciphers: In order to produce a ciphertext, the most recently used counter value is incremented and encrypted, and the result is exclusive-ORed with the plaintext. We notice that the NSA (National Security Agency) Suite B Cryptography approved AES in CTR mode and that AES counter mode is also recommended by NIST (National Institute of Standards and Technology) [20]. E.g., counter mode encryption is used in the IPsec Internet Draft [21] and ATM Security Specification [22].

Before we explain our authentication protocol, we notice that counter mode encryption in dielets has a potential vulnerability in that it can be exploited in a denial-of-service (DoS) attack: An attacker can power up a batch of dielets, causing each dielet to irreversibly increase its counter. This may force counters to run out of range, i.e., the server will not be able to synchronize its own copies of the counter values and will therefore reject future legitimate attempts of dielets to authenticate themselves. In order to prevent significant loss (of many dielets) due to such an efficient *batch-mode* DoS attack, we add a read-out mode before the authentication mode. Only after the dielet confirms (by verifying a challenge) that the smartphone attempts to transmit messages with the dielet, the dielet will enter an authentication mode during which its counter is incremented.

We notice that by adding a read-out mode before the authentication mode, only batch-mode DoS attacks are prevented. I.e., a single-dielet DoS attack is still possible. However, we argue that a successful single-dielet DoS attack only kills a single dielet and does not lead to the significant loss inherent to a batch-mode DoS attack. For this reason, we may treat a single-dielet DoS attack as any other tampering attack against a dielet. Just like the passive sensors that detect tampering attacks, having reached the maximum counter value (if set large enough) detects a single-dielet DoS attack (which is consistent with SHIELD's philosophy).

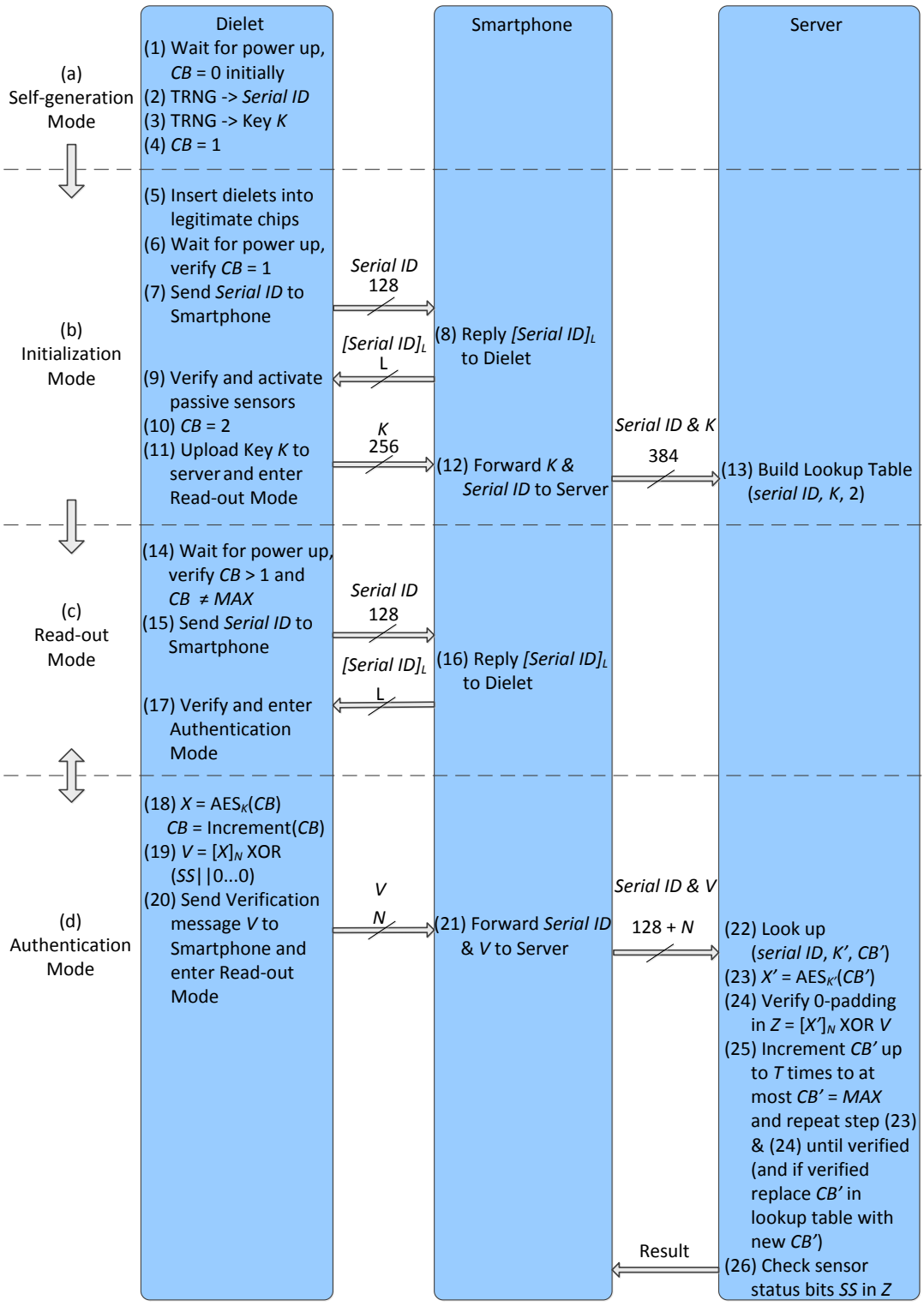

Figure 4: Proposed Authentication and Initialization Protocols for SHIELD. (a) and (b) are the self-generation and initialization modes of the initialization protocol, while (c) and (d) are the read-out and authentication modes of the authentication protocol. Smartphone and server are assumed to be trusted.

3.2 Read-out Mode

Initially, during read-out mode, the dielet waits to be powered up. As soon as it powers up, the dielet checks whether the counter value CB is larger than 1 and less than a maximum value MAX. If the check fails, then either the dielet has not yet been fully initialized (CB equals 0 or 1) and should enter self-generation and initialization mode, or the counter has reached a prefixed maximum number MAX of times the dielet is allowed to enter authentication mode (and during which its sensor status SS is read-out, encrypted and communicated to the server). Checking CB being less than MAX limits the number of encryptions on the dielet and also prevents the counter to roll-over and repeat itself.

If the check passes, then the dielet will proceed to the next step (15). See Figure 4 (c), the dielet transmits its serial ID to the smartphone via the read-out unit. The smartphone truncates the serial ID to L bits and sends the truncated serial ID back to the dielet. The dielet verifies the reply from the smartphone which, if correct (i.e., it corresponds to the dielet's serial ID), puts itself into authentication mode. Because the smartphone has to send the truncated L-bit serial ID back to confirm the communication request, DoS attacks against a batch of dielets *simultaneously* are prevented (since each dielet verifies its own unique L-bit truncated serial ID).

3.3 Authentication Mode

Figure 4 (d) shows the sequence of operations in authentication mode. After the dielet enters authentication mode, the on-board cryptographic key K and the hardware counter block CB are retrieved from NVM to compute $X = AES_K(CB)$. At the same time, the counter block CB is incremented. In step (19) the dielet takes the first N bits of X and exclusive-ORs these with the S-bit status bits (SS) from the dielet's passive sensors padded with zeroes up to N bits ($N \ll 128$). The padded zeros are used for authentication as explained below. The result after the exclusive-OR is an N-bit "verification message" V which is transmitted to the smartphone. After the above steps, the dielet returns to read-out mode, and waits for a next authentication request in step (14).

The smartphone forwards V to the authentication server together with the serial ID. The authentication server looks up the key K' and counter block CB' associated to the serial ID and computes $X' = AES_{K'}(CB')$. The first N bits of X' are exclusive-ORed with V. Without malicious behavior, the result Z should be equal to the string SS padded with zeroes. If Z indeed shows all of the padded zero bits, then the server concludes that the ciphertext was produced by the dielet with the serial ID. In addition, the server concludes that the first S non-zero bits of Z are the sensor status bits of the dielet's passive sensors.

Here, we assume that the smartphone is not compromised in that it can be trusted not to tamper with V (otherwise it can flip some of the first S bits of V which results in maliciously flipping sensor status bits). We argue that in general trust in the smartphone is necessary, since otherwise it can deceive the human user of the smartphone by displaying "authenticated" at the very end of the authentication protocol even if the dielet did not pass authentication. So, having the dielet e.g. encrypt SS padded with zeroes (similar to DARPA's protocol) or digitally sign V (in order to detect flipping of sensor status bits) does not remove the trust requirement of the smartphone. Hence, we make use of the fact that we need to trust the smartphone (at least the SW module that implements the SHIELD protocols): we may use the one-time-padding approach that leads to the construction of V.

The above protocol assumes that the counter block stored at the authentication server CB'

and the dielet's counter block CB are synchronized. This may not be the case in practice due to potential network issues, e.g. disconnection with the server, and for this reason the authentication server repeatedly increases CB' and repeats steps (23) and (24) until either authentication passes (and CB' in the look up table is replaced by the increased CB') or the number of increments goes beyond a pre-fixed threshold T in which case authentication fails (and the server resets the counter block to its originally stored value). After verifying the padded zeros and sensor status bits, the server sends the authentication result to the smartphone. Figure 4 (c) and (d) show the complete authentication protocol.

3.4 Security Analysis

3.4.1 Probability of Successful Impersonation Attack

In our protocol, we use the padded zeros to verify the authenticity of the dielet itself. Therefore the number P of padded zeros is critical to the security of this protocol. Due to the security of AES encryption, an adversarial impersonation of the dielet only has a probability of 2^{-P} to immediately pass authentication (by successfully producing an N-bit ciphertext which corresponds to a message with P padded zeroes under the dielet's secret key). Since we have set a pre-fixed threshold T for increment attempts of CB', the probability of a successful impersonation attack is at most

$$T \cdot 2^{-P}$$

Notice that $P = N - S$, where S is the number of sensor status bits. An appropriate choice of N and S can save energy of the RF transceiver while still achieving sufficient security. E.g., setting $P = 40$ bit zero padding with threshold $T = 8$ yields in the authors' opinion a small enough probability of successful impersonation $\rho = 8 \cdot 2^{-40} = 2^{-37}$: We first notice that $\rho = 2^{-37}$ is not related to the 256-bit security of AES in the following sense. We need the 256-bit security of AES in order to conclude $\rho = 2^{-37}$; if AES would not be this secure, then AES_K of a dielet could possibly be broken in a pre-computation with almost no interaction within a small amount of time by using a very efficient parallelized algorithm (and this would lead to a probability of successful impersonation $\rho = 1$). The security implied by the 256-bit security of AES is not the same as the security implied by $\rho = 2^{-37}$. The 256-bit security of AES means that no feasible algorithm can break AES, while $\rho = 2^{-37}$ means that an impersonation attack can only be successful after $\approx 2^{37}$ authentication mode interactions with the smartphone and server. Clearly, the server is able to detect whether a smartphone sends many different ciphertexts with the same serial ID that do not authenticate: e.g., the server may adopt the policy that after B such failed authentication attempts the server informs the smartphone to simply notify its user that the dielet it is communicating with does not behave normally and that the smartphone will not process any further authentication requests for this serial ID. E.g., by setting $B = 8$, this translates into the property that one out of $2^{37}/8 = 2^{34}$ adversarial impersonations of *different* dielets with a given smartphone is successful. Since 2^{34} is orders more than the total number of dielets that will be produced (within a given generation) and whose serial IDs are recorded at the server, successful impersonation is effectively eliminated.

3.4.2 Probabilistic vs Deterministic Encryption

Because each encryption uses a new counter, CTR Mode AES Encryption is a form of probabilistic encryption, which is much stronger than the deterministic encryption of AES in plain mode which

allows linking ciphertexts that correspond to the same plaintext over time. Even though ciphertexts in AES CTR mode are still deterministically generated by the plaintext and the counter value, an attacker without knowing the counter value is not able to tell whether two ciphertexts correspond to the same plaintext or not.

3.4.3 Try-and-Check Attacks

As a main advantage, our protocol prevents the following attack, which breaks, i.e., *nullifies the effectiveness of DARPA's authentication protocol in that the protocol becomes useless in one very important aspect*: Suppose that a dielet embedded in a legitimate host package is separated from that host package and is inserted into or added to a maliciously created host package of a counterfeit chip or a chip with Hardware Trojan. The dielet is designed such that with significant probability $p > 0$ its passive sensors detect the physical attack that separates it from its host. The adversary, however, may *try* in the hope that the separation from a legitimate chip's host package and insertion into the host package of a counterfeit or malicious chip is not detected by the passive sensors (with probability $1-p$). In DARPA's protocol the adversary can apply the same challenge before and after the physical attack to *check* whether the replied ciphertext has changed or not. If the ciphertext remains unchanged, then this indicates that the physical attack was not detected by the passive sensors which would otherwise have led to a change in the sensor status bits and the ciphertext. So, the attacker is able to find out which of the counterfeit chips will be verified correctly by DARPA's authentication protocol. This means that the attacker is able to figure out which counterfeit or malicious chips can be put into the supply chain without the authentication server being able to detect anything suspicious. This nullifies the effectiveness of the main role[2] of passive sensors in SHIELD: at least some trace or evidence (preferably a fraction of p counterfeited/malicious chips that are being put back into the supply chain) should survive.

On the other hand, if an adversary wants to perform the above attack on a dielet with our authentication protocol, then the adversary is not able to verify whether his attack is successful or not, because counter mode encryption uses a different counter every time which makes encrypted messages look random with respect to one another even if the encrypted messages correspond to the same sensor status bits (here, we notice the fact that AES in CTR mode is a probabilistic and not a deterministic encryption scheme). Therefore the attacker must simply try its luck with putting its counterfeit chips into the supply chain. This means that with significant probability p (the detection capability of the passive sensors) a trace of evidence of the attacker's activities and existence will be detected by the server (once the sensor status bits are transmitted to the server and verified by the server).

3.4.4 Resistance against Non-Invasive Attacks

Extracting a non-expired pair of secret key and serial ID is another way to potentially comprise security. In particular, DPA (Differential Power Analysis) and DFA (Differential Fault Analysis) are two common attacking schemes to recover an on-board secret key of an embedded system.

[2]Besides making it hard (with only a success probability $1-p$) to maliciously extract a small part of a legitimate host package that contains the host package's dielet and add this part to a maliciously assembled host package for a counterfeit chip or a chip with a Hardware Trojan *without the dielet's passive sensors detecting the malicious activities.*

Similar to plain AES encryption, AES CTR mode encryption also suffers from side channel analysis. It only needs 2^{13} power traces to perform a first order DPA against AES in Counter Mode with unknown initial counter [23]. Given the low area budget for dielet design, it is impossible to implement countermeasures against DPA, because countermeasures against DPA usually have more than 5× area overhead [24]. For this reason, DARPA's authentication protocol is vulnerable to differential power analysis [25].

However, our protocol uses counter mode encryption: the counter can also play the role of extra sensor. The dielet itself does not enter authentication mode once the counter reaches a predefined threshold MAX. For example, if the dielet is designed to be read out 100 times during the entire lifetime, this threshold can be set to $MAX = 100$ (plus some more to account for network failures etc.) and an attacker can get at most 100 power traces, far fewer than the number of traces needed to break AES CTR mode encryption: Although the number of traces for a DPA attack depends on the implementation and environment, no one has implemented an attack using 2^8 (our maximum counter value) traces yet. Also, the adversary can only obtain a partial ciphertext from the dielet ($N = 50$ bits in our case), which makes DPA even more difficult.

DFA is another practical attacking approach to recover the cryptographic key of AES [26]. Even if the attacker uses the most powerful differential fault analysis on AES, which only needs one fault to break AES [27], it is still necessary for the attacker to obtain a fault-free ciphertext and a faulty ciphertext that both correspond to the same plaintext. Since the counter in AES CTR mode is irreversibly increasing, the attacker can never get two ciphertexts with the same plaintext.

Here we do notice that [28] proposes a fault attack to break AES CTR mode encryption, however, the used fault model is too precise to be practical: It needs a bit flip fault at the least significant bit of the counter block. Clearly, such a precise fault can not be injected by a clock glitch [29], which always fails the critical path first, because the critical path must be in the implementation of AES rather than in the counter block. Even if the attacker performs reverse engineering and injects faults by laser [30], it is still very unlikely to be able to inject such a precise one-bit fault without flipping other adjacent bits. We conclude that a dielet with an AES CTR mode encryption engine also has immunity against differential fault analysis (as opposed to DARPA's authentication protocol).

In conclusion, an attacker needs to invest in other techniques rather than DPA or DFA to extract serial ID and key pairs from dielets.

3.4.5 Resistance against Invasive and Semi-Invasive Attacks

Invasive and semi-invasive attacks represent another class of security threats. These attacks need to decapsulate the chip. In the dielet, the passive sensors are designed to detect these invasive and semi-invasive attacks. See [2] for a detailed discussion.

We do notice that any faults injected in the counter block after it reaches its maximum value can violate the irreversibility of the counter. This is very dangerous in our protocol, because the attacker can use the reoccurrences of counter values to successfully implement a try-and-check attack or DFA. In order to tackle this attack, we need to take special care of the counter to prevent fault injections with high spatial accuracy [31]. We can implement redundant counters and scatter them across the dielet, but more area-efficient is to implement a sensitive sensor above the counter to detect fault injections [32].

3.4.6 Detection of Number of Dielet Readouts

The counter on the dielet allows the server to learn how many times the dielet has been put into authentication mode when the dielet was offline with respect to the authentication server. This information can be used (besides the passive sensor data) as an additional sensor to record suspicious attacking attempts. Also, as discussed above, our protocol implements a maximum possible counter value MAX after which a dielet will not enter authentication mode any more. This means that even if an attacker obtains a non-expired pair of a serial ID and cryptographic key, the attacker will only be able to use the pair to authenticate at most MAX counterfeit chips (exceeding MAX will also make the authentication server's synchronized counter exceed MAX which prevents authentication of extra chips). This demotivates an economically motivated adversary to invest resources in extracting keys from dielets through hardware reverse engineering, imaging etc.

3.5 Performance Improvement

Compared to DARPA's protocol, our authentication protocol does not only have additional security benefits, but also has improved performance.

Clearly, DARPA's authentication protocol (Figure 3) suffers from three drawbacks. First, the challenge and the sensor status bits are encrypted separately, which doubles the power consumption of the encryption engine. Second, two full 128-bit ciphertexts have to be transmitted from the dielet to the smartphone through the RF transceiver. This results in a long communication latency between the dielet and the smartphone with a large power consumption. Finally, DARPA's protocol needs two full communication rounds between the smartphone and the remote server which due to network latency makes the protocol unnecessarily slow.

(1) Our proposal only needs to encrypt once per authentication request, while DARPA's protocol needs to encrypt the challenge and sensor status bits separately. For this reason, our protocol saves half of the power consumption of AES encryption on the dielet.

(2) Also fewer bits are transmitted between the dielet and the smartphone in our protocol, which makes the transmission more efficient. In step (16) in Figure 4, the smartphone only replies L bits to the dielet to confirm the authentication request ($L = 30$ bits should be enough in practice), instead of sending a 128-bit challenge (which the dielet interprets as a plaintext for encryption). In our protocol the value of the counter block CB is in essence used as a built-in challenge. In step (20) in Figure 4, the dielet only sends N bits to the smartphone.The total number of bits transmitted between the dielet and the smartphone is therefore $128 + L + N$. We recommend $L = 30$ and $N = S + P = 10 + 40 = 50$ (assuming at most 10 different sensors on board of the dielet), which results in only 208 transmitted bits. This is more than twice less than the 448 transmitted bits in DARPA's protocol. (Notice that improvement is based on our suggested L and N, so it may differ for different choices of L and N.)

(3) Finally, the computation time or transmission time between the dielet and the smartphone is in microseconds, but the network communication latency is on the order of tens of milliseconds (e.g. the Boston to Santa Clara round trip network latency is larger than 86 ms [6]). Hence, it is important to minimize the number of communication rounds between the smartphone and the server. Since only one full communication round is needed between the smartphone and the server in our protocol, the entire authentication can complete much faster (SHIELD requires that the authentication protocol finishes in 2 seconds [7]).

3.6 Other Design Considerations

3.6.1 AES CTR Mode vs One-Time Pad

Since CTR Mode allows preprocessing, we can use an AES CTR Mode generated one-time pad (OTP) to replace the key storage and encryption engine on the dielet chip. Hence the overhead of this alternative heavily depends on the number of authentications in the entire lifetime of this dielet. For example, if $N = 50$, and dielets are supposed to be read out 100 times in their lifetimes, then 5000 bits have to be fused into the dielet's NVM during manufacturing. Compared to the 320 bits which need to be written in DARPA's protocol, this causes an unacceptable 15× longer initialization time.

3.6.2 AES CTR Mode vs Stream Cipher

Basically, counter mode turns a block cipher into a stream cipher. The advantage of using a stream cipher instead of AES CTR Mode is low area overhead and high throughput, because stream ciphers are designed to be used in a constrained environment for lightweight applications. A first concern is that no stream cipher has been approved or recommended by NIST and NSA. Second, the dielet is passively powered up, therefore the power signal is controlled by the adversary. For this reason the states of the stream cipher need to be stored into NVM every clock cycle. This requires a large sized circuit for accessing state bits in NVM as well as a large amount of power consumption for updating each bit in NVM every clock cycle [33].

3.6.3 Irreversible Counter vs TRNG

The next section describes an initialization protocol which uses a TRNG for generating a key and serial ID. It seems that area can be saved if, instead of a counter, a TRNG-generated random nonce is used in AES CTR mode. Similar to an irreversible counter, a random nonce also provides the one-time character needed to thwart the "try-and-check" attack. Notice that the use of random nonces not only eliminates the counter from the design, but also eliminates the control logic which implements the read-out mode (since the use of random nonces avoids the batch-mode DoS attack).

However, in order to efficiently (without much area overhead) prevent successful DPA attacks, dielets need to limit the number of times they can been queried. Hence, a counter is needed to count the number of times a random nonce has been generated and as soon as the counter reach some maximum threshold value, the dielet should stop responding. This means that the dielet again needs to prevent batch-mode DoS attacks by adding read-mode control circuitry, i.e., no area is saved. If we want to keep the same security guarantees, then there is no advantage of using a TRNG-generated random nonce in AES CTR mode.

3.6.4 Integration with RFID tags in Supply Chain Management

In current SCM (Supply Chain Management), RFID (Radio Frequency Identification) tags are used as EPCs (Electronic Product Codes) to track products in the supply chain [34]. A dielet inserted in the host package of a chip can besides having the required SHIELD functionality also implement EPC functionality. The dielet can use the RF channel to communicate messages to a smartphone or RFID reader and this allows a seamless integration of SHIELD and EPC functionality in dielets. In both cases the same centralized remote server can collect the sensor status bits as well as tracking information of chips in the supply chain.

4 Initialization Protocol

Dielets need to be initialized with their own unique serial IDs and keys, and also the authentication server needs to store a copy of these serial ID and key pairs in its own database. Finally, an initialization process should activate dielets in that their passive sensors will start recording tampering events in their sensor status bits.

Clearly, the passive sensors of a dielet should not be enabled before the dielet is inserted into a host package of a legitimate IC (otherwise, the passive sensors register the insertion as a tampering event). One (insecure) solution is to execute a complete initialization protocol from beginning to end including activation of passive sensors[3] right after a dielet is inserted into its host package. This solution implies that not-yet-initialized dielets are in transit from one trusted (dielet-fabrication/assembly) facility to another trusted (dielet-insertion) facility and such dielets can potentially be intercepted. An adversary can tamper with a dielet in the host package of a legitimate IC in order to extract its serial ID and key pair, next take an intercepted dielet and insert it into the host package of a counterfeit IC or IC with Hardware Trojan, and initialize the intercepted dielet with the extracted serial ID and key pair.

To remedy the protocol, the initialization process should have at least two phases/modes. During the first initialization phase dielets get their own serial ID and key pairs while they are in the trusted dielet fabrication and assembly facility and preferably while they are still on the wafer. During the second initialization phase the passive sensors of a dielet are being activated so that from then onwards they can record detected tampering events in the sensor status bits; the second initialization phase should happen in the trusted IC assembly facility right after a dielet's insertion into the host package of a legitimate IC.

Rather than fusing serial IDs and cryptographic keys directly into each dielet while they are still on the wafer[4], we propose to have each dielet self-generate its key and serial ID *in parallel* by exploiting on-chip randomness by using a TRNG while they are still on the wafer at the trusted dielet fabrication and assembly facility.

4.1 True Random Number Generator

A TRNG (True Random Number Generator) is a hardware security primitive, which harnesses the on-chip noise to generate a random number for security applications. A good TRNG design relies on a good entropy source, a decent harvesting mechanism and a postprocessing mechanism [36]. A good entropy source should provide as much entropy as possible, while a decent harvesting mechanism should be able to collect as much entropy as possible from the entropy source. The postprocessing mechanism is not necessary in every TRNG design, but it strengthens the TRNG design and eliminates the bias in the output bits. E.g., we can implement a von Neumann extractor [37],

[3]It is outside the scope of this paper to develop an insertion technology which keeps passive sensors fresh in that they keep on being capable of detecting (with a significant probability) the first future tampering event (the sensor status bits only need to record whether a tamper event has happened and do not need to record how many have happened). Especially, if a sensor is using e.g. photo sensitive material, then this becomes a challenge.

[4]One solution is to add extra power lines and access circuitry on the wafer itself, which increases fabrication costs. Another solution is to use RF communication to sequentially write and fuse each unique serial ID and key. This leads to a non-parallelized approach and a simple calculation assuming one million $0.01mm^2$ dielets on a single wafer (dielets are required to fit inside a $0.01mm^2$ area [7]) and assuming a maximum rate of fusing 2500 dielets per second (using high performance equipment [35]) show that initialization may take an impractical 7 minutes via the RF channel.

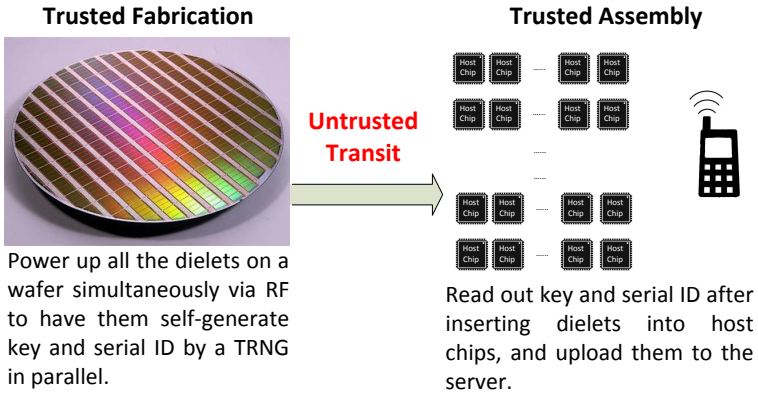

Figure 5: Our proposed initialization process.

which is a simple finite state machine for extracting a non-biased true random number from a biased entropy source. The idea is to repeatedly measure two consecutive output bits of the TRNG until a 01 or 10 is measured. A 01 is interpreted as a 0 and a 10 is interpreted as a 1. Since a 01 and 10 are equally likely, their 0 and 1 bit interpretation is un-biased.

We propose to use the SRAM-based TRNG in [38] because it harvests randomness of the noise generated by an area efficient metastable structure which is just an SRAM cell. This means that a dielet only needs one SRAM cell structure to implement a TRNG (which is repeatedly evaluated in order to generate a serial ID together with a cryptographic key).

4.2 Initialization Protocol

Figure 4 (a) shows the self-generation mode of our proposed initialization protocol: The manufacturer can power up all the dielets in one wafer simultaneously by using a large power RF antenna. This enables the self-generation of a key and a *random* serial ID (which are fused once generated) in each dielet in parallel. The parallel generation and storage of keys and random serial IDs in the dielets' NVMs takes several microseconds. The counter block CB is initialized to zero, and set to one after the self-generation mode is finished.

Once all the dielets on the wafer generated and stored their serial IDs and keys, they are ready to leave the trusted dielet fabrication and assembly facility. When they arrive at the trusted IC assembly facility, the dielets are inserted into the host package of legitimate ICs and complete the initialization mode of our proposed initialization protocol, see Figure 4 (b): This mode builds the server's database by separately reading the cryptographic key and random serial ID of each dielet as part of the dielet insertion process and activation of passive sensors (note that the server may in addition associate a unique serial ID in standard format to each random serial ID in its data base). The counter block CB is incremented to 2. Figure 5 depicts the entire initialization process and threat mode.

We notice that even if an attacker intercepts a dielet (which is in transition between the self-generating mode and initialization mode) and inserts it into the host package of a counterfeit IC or IC with Hardware Trojan, the authentication server will not authenticate the malicious IC since its database does not recognize the serial ID and key.

4.3 Serial ID Collision

Since a randomly generated serial ID cannot guarantee uniqueness, the length of the serial ID needs to be adapted in order to make the probability of a serial ID collision negligible. Let λ be the probability of a collision among all generated serial IDs, n be the number of bits of a serial ID, and t be the (maximum) number of dielets we plan to produce. From [39], we know that λ, n, and t are related in the following way:

$$t \approx 2^{(n+1)/2} \cdot \sqrt{\ln\left(\frac{1}{1-\lambda}\right)}.$$

If we choose $t = 2^{44.5} = 2.5 \cdot 10^{13}$, and $\lambda = 2^{-40}$, then the size of a serial ID must be $n = 128$ bits. In DARPA's protocol [7], the serial ID has only 64 bits. As explained in Figure 4, even though the random serial IDs in our protocols require $n = 128$ bits, still only 208 bits are transmitted between the dielet and smartphone in our authentication protocol. This improves over DARPA's authentication protocol which needs to transmit twice more bits (448 bits).

4.4 Performance

Because we use a true random number generator to allow each dielet to generate its own serial ID and key, dielets are able to perform this computation in parallel (without any extra equipment). The time to self-generate and fuse keys and serial IDs in parallel is negligible per wafer and makes our initialization protocol the first practical solution.

5 Implementation

For our protocols we implemented the extra control logic in 32nm technology [40] (including an SRAM-based TRNG with a von Neumann extractor, an 8-bit counter, and a new state machine with four states corresponding to the four modes of the dielet) and made a comparison with the control logic for DARPA's protocol. In our implementation, we consider the interaction with the transceiver, sensor status bits and NVM as primary input/output to our control logic. In our implementation, we used an SRAM-based TRNG which only uses one SRAM cell to harvest run-time physically random noise [38], together with a von Neumann extractor [37] as post-processing circuitry. In order to save area on the dielet, all operations in our implementation are byte-wise, including a compact 8-bit datapath AES-256 encryption-only core based on the architecture in [41], which instantiates two S-box implementations (one for datapath of state and the other one for key expansion) and takes 224 clock cycles to complete one AES-256 encryption. In this AES encryption engine, we used an S-box implementation proposed in [42], which first changes the basis of the input value from GF(2^8) to a composite field GF(2^8)/GF(2^4)/GF(2^2), and computes all of the operations in this composite field before converting the basis back in the end. Because Canright did a thorough investigation on S-box implementation, this is still the optimal (smallest) S-box implementation. Although we did not implement the other components on the dielet, we demonstrate the complexity of control incurred by our protocol by comparing the additional area with the area of AES-256. We notice that the area of AES-256 takes 55% of the allowed area of the dielet in 32nm technology ($0.01mm^2$ [7]). The control logic of DARPA's authentication protocol costs only 2% area of the dielet, while the control logic of our protocols is 6% giving an area overhead in control logic of only 4%.

Also, the area of an extra 64-bit NVM (for storing a 128-bit random serial ID compared to a 64-bit serial ID in DARPA's protocol) can be estimated by multiplying the cell size of each bit by 64. Because the cell size of NVM varies from $4F^2$ to $22F^2$, where F is the feature size, the area of a 64-bit NVM is negligible (<0.01% in 32nm technology) [43].

The passive sensors are mostly deployed as an additional layer above the circuit, so it increases the thickness of the dielet [2]. In addition, another observation is that the existing RF transceiver or antenna cannot fit the SHIELD area requirement at all, because the size of an RF antenna is proportional to the wavelength of the RF signal [44]. Currently, the smallest antenna record is held by two researchers at BIT (Mesra) Ranchi, India developed in 2013 with size of 14mm × 11mm [45]. Since the size of the smallest antenna is still far away from DARPA's requirement ($0.01mm^2$), the SHIELD dielet should not communicate via conventional RF technology, and the antenna design for SHIELD is still an open question. This statement is also confirmed in DARPA's Call For Proposal [7].

Compared with the area of AES, passive sensors , RF transceiver, possibly a Built In Self Test (BIST) circuitry [46], and original NVM on the dielet, our additional area utilization (4%) is very small.

6 Conclusion

This paper clearly demonstrates the superiority of AES in CTR mode encryption over plain AES encryption. First basing the SHIELD authentication protocol on AES in CTR mode results in dramatic performance improvements with respect to a 2× reduction in the number of communication rounds with the server which speeds up authentication by approximately a factor 2 (due to the relatively large network latency) as well as a 2× reduction in number of bits transmitted between dielet and smartphone and a 2× reduction in number of required AES encryptions leading to a significant decrease in power consumption. These performance improvements allow a dielet design that meets the heavily constrained SHIELD specifications with respect to area overhead, power consumption, and speed.

However, much more important is the security improvement offered by AES in CTR mode: Using plain AES only offers deterministic symmetric key encryption allowing an adversary to link ciphertexts over time. This leads to the introduced *try-and-check attack* on DARPA's suggested authentication protocol. The attack *nullifies a main projected benefit of SHIELD* since an adversary is able to eliminate any trace/evidence of his activities that can be detected by dielet sensors. It is of crucial importance to make sure security engineers understand when and how to use AES in CTR mode encryption, in particular for a product as important as SHIELD-dielets which is supposed to create a trusted foundation for embedded systems by restoring trust in outsourced IC fabrication and assembly with respect to supply chains that are out of one's own control. Besides preventing the try-and-check attack, AES in CTR mode also offers plenty of other security benefits: the counter may serve as an additional indicator of suspicious behavior, the counter can be used to limit the lifetime of dielets which in turn prevents (non-invasive) DPA and DFA attacks.

As a second contribution we introduce the first secure and practical initialization protocol for dielets. The main insight is to have each dielet self-generate its serial ID and key using a TRNG (which can be implemented using one SRAM cell).

Meanwhile, the additional area overhead of our protocol on top of the area overhead of DARPA's authentication protocol (which excludes initialization) is only 4% of the dielet area size, which is

small compared with the required area of AES, passive sensors, RF transceiver, possibly BIST circuitry and NVM on the dielet.

Acknowledgment

The authors would like to thank Prof. Mark M. Tehranipoor and Prof. Domenic Forte for the fruitful discussions, when we developed these protocols. Moreover, the authors would like to thank the valuable comments from anonymous reviewers at CHES 2015.

References

[1] SEMI, "White paper: IP infringement causes $4 billion loss to industry annually," http://www.semi.org/en/Press/P043775, 2008.

[2] D. Shahrjerdi, J. Rajendran, S. Garg, F. Koushanfar, and R. Karri, "Shielding and securing integrated circuits with sensors," in *Computer-Aided Design (ICCAD), 2014 IEEE/ACM International Conference on*. IEEE, 2014, pp. 170–174.

[3] AES, NIST, "Advanced Encryption Standard," *Federal Information Processing Standard, FIPS-197*, vol. 12, 2001.

[4] H. Lipmaa, D. Wagner, and P. Rogaway, "Comments to NIST concerning AES modes of operation: CTR-mode encryption," 2000.

[5] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 31, no. 7, pp. 994–1007, 2012.

[6] K. D. Bowers, M. van Dijk, A. Juels, A. Oprea, and R. L. Rivest, "How to tell if your cloud files are vulnerable to drive crashes," in *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 2011, pp. 501–514.

[7] DARPA, "Supply chain hardware basic concepts and taxonomy of dependable and secure computing," *Microsystems Technology Office/MTO Broad Agency Announcement*, 2014.

[8] K. Huang, J. M. Carulli, and Y. Makris, "Counterfeit electronics: A rising threat in the semiconductor manufacturing industry," in *Test Conference (ITC), 2013 IEEE International*. IEEE, 2013, pp. 1–4.

[9] U. Guin, D. DiMase, and M. Tehranipoor, "Counterfeit integrated circuits: detection, avoidance, and the challenges ahead," *Journal of Electronic Testing*, vol. 30, no. 1, pp. 9–23, 2014.

[10] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," 2010.

[11] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Silicon physical random functions," in *Proceedings of the 9th ACM conference on Computer and communications security*. ACM, 2002, pp. 148–160.

[12] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, "Design and implementation of PUF-based" unclonable" RFID ICs for anti-counterfeiting and security applications," in *RFID, 2008 IEEE International Conference on.* IEEE, 2008, pp. 58–64.

[13] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security." in *USENIX Security*, 2007, pp. 291–306.

[14] G. K. Contreras, M. T. Rahman, and M. Tehranipoor, "Secure split-test for preventing ic piracy by untrusted foundry and assembly," in *Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2013 IEEE International Symposium on.* IEEE, 2013, pp. 196–203.

[15] R. W. Jarvis and M. G. McIntyre, "Split manufacturing method for advanced semiconductor circuits," Mar. 27 2007, uS Patent 7,195,931.

[16] X. Zhang and M. Tehranipoor, "Design of on-chip lightweight sensors for effective detection of recycled ICs," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 22, no. 5, pp. 1016–1029, 2014.

[17] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *Security and Privacy, 2007. SP'07. IEEE Symposium on.* IEEE, 2007, pp. 296–310.

[18] Y. Jin and Y. Makris, "Hardware trojan detection using path delay fingerprint," in *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on.* IEEE, 2008, pp. 51–57.

[19] M. Bushnell and V. D. Agrawal, *Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits.* Springer Science & Business Media, 2000, vol. 17.

[20] M. Dworkin, "Recommendation for block cipher modes of operation. methods and techniques," DTIC Document, Tech. Rep., 2001.

[21] R. Housley, "Using AES counter mode with IPsec ESP," *IPsec Working Group, Internet Draft, RSA Laboratories,(Jul. 2002)*, 2003.

[22] ATM Forum Technical Committee, "ATM security specification version 1.0," *af-sec-0100.001, February*, 1999.

[23] J. Jaffe, *A first-order DPA attack against AES in counter mode with unknown initial counter.* Springer, 2007.

[24] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, "Pushing the limits: a very compact and a threshold implementation of AES," in *Advances in Cryptology–EUROCRYPT 2011.* Springer, 2011, pp. 69–88.

[25] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology - CRYPTO 99.* Springer, 1999, pp. 388–397.

[26] X. Guo, D. Mukhopadhyay, C. Jin, and R. Karri, "Security analysis of concurrent error detection against differential fault analysis," *Journal of Cryptographic Engineering*, pp. 1–17, 2014.

[27] M. Tunstall, D. Mukhopadhyay, and S. Ali, "Differential fault analysis of the advanced encryption standard using a single fault," in *Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication.* Springer, 2011, pp. 224–233.

[28] R. Tirtea and G. Deconinck, "Specifications overview for counter mode of operation. security aspects in case of faults," in *Electrotechnical Conference, 2004. MELECON 2004. Proceedings of the 12th IEEE Mediterranean*, vol. 2. IEEE, 2004, pp. 769–773.

[29] M. Agoyan, J.-M. Dutertre, D. Naccache, B. Robisson, and A. Tria, "When clocks fail: On critical paths and clock faults," in *Smart Card Research and Advanced Application.* Springer, 2010, pp. 182–193.

[30] F. Courbon, P. Loubet-Moundi, J. J. A. Fournier, and A. Tria, "Increasing the efficiency of laser fault injections using fast gate level reverse engineering," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014, Arlington, VA, USA, May 6-7, 2014*, 2014, pp. 60–63.

[31] S. P. Skorobogatov and R. J. Anderson, "Optical fault induction attacks," in *Cryptographic Hardware and Embedded Systems-CHES 2002.* Springer, 2003, pp. 2–12.

[32] J. O. Chu, G. M. Fritz, H. J. Hovel, Y.-H. Kim, D. Pfeiffer, and K. P. Rodbell, "Integrated circuit tamper detection and response," Oct. 14 2014, uS Patent 8,861,728.

[33] V. Mohan, T. Bunker, L. Grupp, S. Gurumurthi, M. R. Stan, and S. Swanson, "Modeling power consumption of NAND Flash memories using flashpower," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 32, no. 7, pp. 1031–1044, 2013.

[34] S. E. Sarma, S. A. Weis, and D. W. Engels, "RFID systems and security and privacy implications," in *Cryptographic Hardware and Embedded Systems-CHES 2002.* Springer, 2003, pp. 454–469.

[35] AMS, "AS3953A datasheet - ams," http://ams.com.

[36] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *Computers, IEEE Transactions on*, vol. 56, no. 1, pp. 109–119, 2007.

[37] J. Von Neumann, "13. various techniques used in connection with random digits," 1951.

[38] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up sram state as an identifying fingerprint and source of true random numbers," *Computers, IEEE Transactions on*, vol. 58, no. 9, pp. 1198–1210, 2009.

[39] B. Preneel, C. Paar, and J. Pelzl, *Understanding cryptography: a textbook for students and practitioners.* Springer, 2009.

[40] Synopsys, "32/28nm generic library," http://www.synopsys.com.

[41] P. Hamalainen, T. Alho, M. Hannikainen, and T. D. Hamalainen, "Design and implementation of low-area and low-power AES encryption hardware core," in *Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006. 9th EUROMICRO Conference on.* IEEE, 2006, pp. 577–583.

[42] D. Canright, "A very compact s-box for AES," in *Cryptographic Hardware and Embedded Systems–CHES 2005*. Springer, 2005, pp. 441–455.

[43] D. S. Jeong, R. Thomas, R. Katiyar, J. Scott, H. Kohlstedt, A. Petraru, and C. S. Hwang, "Emerging memories: resistive switching mechanisms and current status," *Reports on Progress in Physics*, vol. 75, no. 7, p. 076502, 2012.

[44] V. Chawla and D. S. Ha, "An overview of passive RFID," *Communications Magazine, IEEE*, vol. 45, no. 9, pp. 11–17, 2007.

[45] S. Pal and M. Chakraborty, "Super compact planar microstrip antenna for ultra wide band applications," 2013, patent ID: KOL/814/2013.

[46] E. J. McCluskey, "Built-in self-test techniques," *Design & Test of Computers, IEEE*, vol. 2, no. 2, pp. 21–28, 1985.