New Applications of Physical Unclonable Functions

Rainer Falk and Steffen Fries
Corporate Technology
Siemens AG
Munich, Germany
e-mail: {rainer.falk|steffen.fries}@siemens.com

***Abstract*—Physical Unclonable Functions (PUF) realize the functionality of a "fingerprint" of a digital circuit. They can be used to authenticate devices without requiring a cryptographic authentication algorithm, or to determine a unique cryptographic key based on hardware-intrinsic, device-specific properties. It is also known to design PUF-based cryptographic protocols. This paper presents several new applications of PUFs. They can be used to check the integrity, or authenticity of presented data. A PUF can be used to build a digital tamper sensor or a digital degradation sensor. An identifying information in a communication protocol can be determined using a PUF, or a licensing mechanism can be realized. The bootstrapping of cryptographic credentials can be protected by a PUF.**

Keywords–physical unclonable function; key extraction; embedded security; licensing; configuration integrity

I. INTRODUCTION

The need for technical information technology (IT) security measures increases rapidly to protect products and solutions from manipulation and reverse engineering. Cryptographic IT security mechanisms have been known for many years, and are applied in smart devices (internet of things, industrial and energy automation, operation technology). Such mechanisms target authentication, system and communication integrity and confidentiality of data in transit or at rest.

Critical Infrastructures (CI) and especially cyber security in critical infrastructures has gained more momentum over the last years. The term "critical infrastructure" in the context of this paper is used to describe technical installations, which are essential for the functioning of the society and economy of a country, but also globally. Typical critical infrastructures in this context are the smart energy grid (including central or distributed energy generation, transmission, and distribution), water supply, healthcare, transportation, telecommunication services, just to state a few. The increased threat level becomes visible, e.g., through reported attacks on critical infrastructure, but also through legislation, which meanwhile explicitly requires the protection of critical infrastructures and reporting about serious attacks.

Information Technology (IT) security in the past was addressed mostly in common enterprise IT environments, but there is a clear trend to provide more connectivity to operational sites, which are quite often part of the critical infrastructure. Examples for operational sites are industrial automation or energy automation. This increased connectivity leads to a tighter integration of IT and Operational Technology (OT). IT security in this context evolves to cyber security to underline the mutual relation between the security and physical effects.

One essential basis for the operation of security mechanisms is typically a cryptographic key that has to be stored securely on devices. A significant effort is often required in practical realizations to protect the storage of cryptographic keys, e.g., by securely integrating external or internal hardware integrated circuits (IC). In current security research on PUFs, methods are investigated that directly use a unique physical property of an object as a physical fingerprint. The problem addressed in this paper is the practical application of physical unclonable functions (PUF) in new, innovative ways, being an extended version of [1]. Upcoming industrial security standards for industrial automation and control systems as IEC 62443 [2] require explicitly a hardware-bound storage for cryptographic keys. A hardware-bound key store or hardware trust anchor can be realized by using a separate security integrated circuit (IC), or by using in integrated hardware trust anchor of a microcontroller. In both cases, a dedicated hardware security functionality has to be realized.

Small random differences of physical properties are used by a PUF to identify an object directly, or to derive a cryptographic key for conventional cryptographic IT security mechanisms [3]. A digital circuit, i.e., a semiconductor integrated circuit, can contain a digital circuit element called a PUF to determine the physical device fingerprint. Minimal differences in the semiconductor structure, like for instance the doping of a semiconductor, the layer thickness, or the width of lines arise at the production randomly. This is similar to the random surface structure of paper sheets. These chip individual properties are "simply there" without being designed-in explicitly, or being programmed by a manufacturer during production. Such a device fingerprint shall be unique, and not be reproducible easily (unclonability). In addition, the fingerprint can be modified, or even destroyed when the IC is manipulated physically. A PUF can be used as simple low-cost alternative to a dedicated hardware key store security circuit, or as additional protection mechanism to conventional cryptographic security mechanisms. It may be useful for simple devices in the Internet of Things to bootstrap cryptographic security credentials using an intrinsic device-specific fingerprint to protect the bootstrapping process. Much research has been

spent on constructions for realizing a PUF, and for extracting a cryptographic key from a PUF [4][5][6][7].

After giving an overview of some major realization possibilities for a digital PUF in Section II, basic usages of a PUF are summarized in Section III. The main contribution of the paper is in Section IV, describing several new applications of PUF technology. Section VI concludes with a summary, and an outlook.

II. Physical Unclonable Functions as Digital Device Fingerprint

A PUF can be realized on a semiconductor circuit to determine a device-specific piece of information depending on variations in the target physics due to the manufacturing process. The information provided by the PUF can be used directly for low-cost authentication, to determine a serial number as an identifier, or as cryptographic key. The semiconductor circuit can be an application-specific integrated circuit (ASIC), or a field-programmable gate array (FPGA). This section gives a short overview about PUFs. More detailed information is available in tutorials on PUFs [4][5][6][7].

PUFs have been a major topic of academic research. However, PUF technology is already applied commercially. Examples are Intrinsic ID [8], Verayo [9][10], Microsemi Smartfusion2 FPGAs [11], and NXP smart card ICs [12].

Common digital circuits are designed to provide identical behavior on different ICs. However, a PUF circuit is designed to provide different results on different ICs, but identical or at least similar results on the same IC when the function is executed repeatedly.

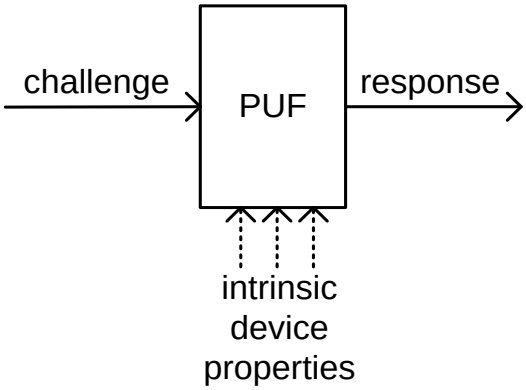

Figure 1. Challenge-Response-PUF

Figure 1 shows a challenge-response PUF, in which the PUF circuit determines a response value depending on a provided challenge value. Weak PUFs and strong PUFs are distinguished: while a strong PUF has a wide range of challenge input values, a weak PUF has no, or only a very limited set of challenge values. A strong digital PUF can be realized by reconfiguring a digital PUF circuit depending on the challenge value.

A PUF performs a computation to determine a response value depending on a given challenge value. Intrinsic device properties influence the PUF calculation so that the calculation of the response is different on different devices, but reproducible – with some bit errors – on the same device.

The objective of a PUF circuit is that on the same IC, the response value for a given challenge value is stable (reproducibility), while on different ICs, the response values are different (uniqueness). As binary values are used for challenge and response values, the similarity can be measured by the Hamming weight, i.e., the number of different bits. The measure for reproducibility is the intra-device Hamming distance, i.e., the mean value of the number of different bits when the PUF is executed multiple times for a given challenge value. The measure for the uniqueness is the inter-device Hamming distance, i.e., the mean value of the number of different bits when executed in different ICs.

Figure 2 shows three examples of well-known constructions of PUFs and their mechanical analogon:

- SRAM-PUF: power-up value of static random-access memory (SRAM) cells
- RO-PUF (Ring oscillator PUF): oscillator frequency
- Arbiter PUF: time delay

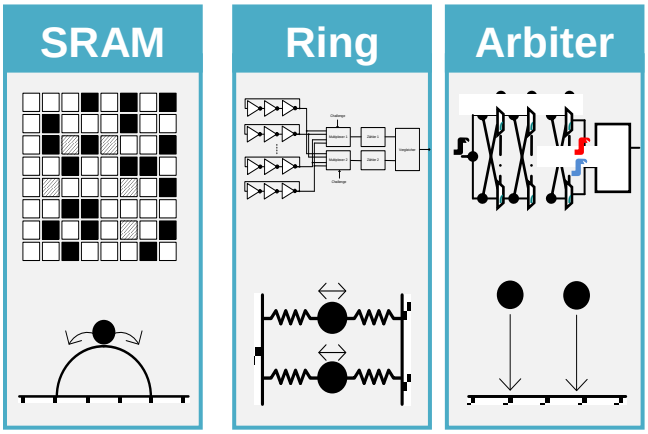

Figure 2. Example PUF Realizations and Their Mechanical Analogon

Many more constructions for a digital PUF have been proposed, e.g., bi-stable Ring PUF, Flip-Flop-PUF, Glitch PUF, Cellular Non-linear Network PUF, or Butterfly-PUF.

A. SRAM-PUF

A digital memory can store binary values 0 and 1. After power-up, some memories show a device-specific initialization pattern. The power-up value of a memory cell can be either 0 or 1, or being instable (sometimes 0, sometimes 1). The pattern of power-up values of its memory cells is characteristic of a memory IC, depending on small variations of the semiconductor physics of each memory cell.

A mechanical analogon for the power-up is a ball placed on the top of a hill [13]. When the whole geometry is exactly symmetric, the ball will roll-down to the left side and to the right side with the same probability. If the hill, or the ball, would have some asymmetries from manufacturing, the ball will tend to roll-down either to the left side or to the right side.

B. Ring Oscillator PUF (RO-PUF)

A digital circuit can realize an oscillator using a delay circuit with a feedback loop (ring oscillator). The oscillation frequency depends on manufacturing variations. The frequency of two identically designed oscillators can be compared using a counter, and comparator. Depending on the IC, one or the other will oscillate with a higher frequency. Realizing multiple oscillators, a "fingerprint" of the digital circuit can be obtained.

A mechanical analogon is an oscillating mass, and spring. Two identical physical realizations will in practice have a slightly different oscillation frequency, depending on small physical variations.

C. Arbiter PUF

A further effect that can be used to build a PUF is time delay. Two identically designed signal paths will show minimal differences in the respective delay. After giving in input signal to both signal paths at the same time, an arbiter circuit determines the faster signal path, i.e., the signal path on which the signal appears first,

A mechanical analogon is a drop test for two identically manufactured masses. Depending on variations in the height, or the surface of the masses, one will tend to impact first on the floor.

III. Basic PUF Applications

A PUF can be used for security purposes in different ways. It can be used as low-cost object authentication, or to determine a cryptographic key. This section describes these two basic applications, and gives examples for some specific usages of PUFs.

A. Object Authentication

Authentication is an elementary security service proving that an entity in fact possesses a claimed identity. Often natural persons are authenticated. The basic approaches a person can use to prove a claimed identity are by something the person knows (e.g., a password), by showing something the person has (e.g., passport, authentication token, smart card), or by exposing a physical property the person has (biometric property, e.g., a fingerprint, voice, iris, or behavior).

Advanced authentication techniques make use of multiple authentication factors, and performing authentication continuously during a session. With multi-factor authentication, several independent authentication factors are verified, e.g., a password and an authentication token. With continuous authentication, also called active authentication, the behavior of a user during an authenticated session is monitored to determine if still the authenticated user is using the session.

With ubiquitous machine-oriented communication, e.g., coming with the Internet of Things and interconnected cyber physical systems, also devices have to authenticate in a secure way. Considering the threat of counterfeited products (e.g., consumables, replacement parts) and the increasing importance of ubiquitous machine-based communication, also physical objects need to be authenticated in a secure way. Various different technologies are used to verify the authenticity of products, e.g., applying visible and hidden markers, using security labels (using, e.g., security ink or holograms), and by integrating cryptographic authentication functionality in wired product authentication tokens, or Radio Frequency Identification (RFID) authentication tags.

An object or digital circuit can be identified by a serial number. For authentication, a cryptographic authentication protocol can be used, requiring a secret/private key to be available on the object to be authenticated.

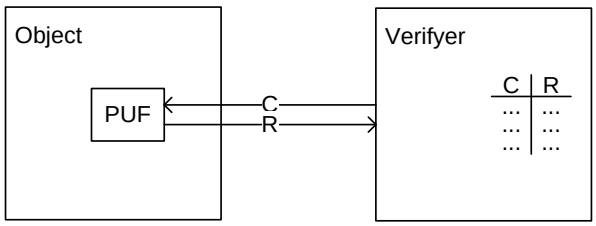

Figure 3. Challenge-Response-Authentication

For authentication, a challenge value is sent to the object to be authenticated. A corresponding response value is sent back and verified. The response is determined by the PUF. As only an original product can determine the correct response value corresponding to a challenge, the product entity or a dedicated part of the product is thereby authenticated.

Figure 3 shows how an object becomes authenticated by a verifier. The verifier maintains a database of reference challenge response pairs. For example, the database was filled during production of the object by recording arbitrary challenge-response-pairs. During the authentication the verifier selects a challenge value of the database and sends it to the object to be authenticated. The response value R is determined by means of the PUF, and transferred back to the verifier. The verifier compares the received response value with the reference value stored in the database. If these are similar, i.e., the number of different bits does not exceed a threshold, the object is authenticated successfully.

B. Cryptographic Key Extraction

A cryptographic key can be determined based on inexact, noisy data. A "fuzzy key extractor" is a functionality that determines a stable cryptographic key using a PUF, and helper data [14][15]. The helper data allows to correct bit errors of responses (noisy data), and to map the PUF output to a given cryptographic key. A main advantage is that no secure non-volatile memory is needed on the device to store a cryptographic key.

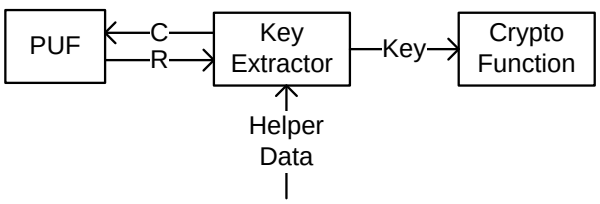

Figure 4. PUF Key Extraction

The PUF is used internally within a digital circuit to determine response values, see Figure 4. The helper data does not have to be stored securely. It can be used only by a single IC to determine the cryptographic key on the device.

C. Further PUF Applications

Several further applications, besides the two basic PUF usages outlined above, have been proposed and designed. The following list section gives an overview on related work.

- A PUF can be used to prevent utilizing specific features of semiconductor ICs. Without chip-specific aiding information, the performance of an IC is reduced or access to certain memory partitions is prevented. Also, a PUF can be used to bind software intellectual property to a FPGA device by encrypting the software code using a PUF-generated device key [16], which is typically done during manufacturing. This solution can be used to protect for instance remote software updates [17].
- A PUF can be used to protect the execution of software code: the Control Flow Graph of an executed program depends on the output of a PUF [18].
- It is known to include a measurement value determined by a sensor as part of the challenge of a PUF to authenticate the sensor measurement [19][20]. This allows authenticating sensor measurements.
- A PUF can be used also in data communication to determine a message integrity checksum (message integrity code, message authentication code) [21]. While a real, physical PUF is used to determine the message authentication code by the sender, a simulated, algorithmic model of the PUF is used to verify the checksum by the receiver.
- Furthermore, the cryptographic key derived by a PUF of a semiconductor can be used to decrypt configuration data [22].
- A PUF can, as security primitive, be integrated in a cryptographic protocol directly [23][24].

D. Limitations of PUF

Building security solutions using PUFs, it is important to understand their limitations. Important issues to be considered are:

- Attacks on the PUF itself, and attacks to PUF support functions as a fuzzy key extractor need to be taken into account. This relates for instance to the PUF model building, to physical attacks on PUFs, and to side channel and fault injection attacks [24][25].
- Robustness of a PUF implementations with respect to tampering, e.g., how vulnerable is a solution in fact against attacks on the opened chip, using e.g., focused ion beams.
- Reliability of the PUF with respect to the long term application in devices related to ageing, and environmental conditions as temperature and electromagnetic radiation.
- Required processes for enrollment of helper data and security management data, which relates on one hand to data on the PUF device, and to data maintained within backend security systems. On the other hand, depending on the PUF application the handling of the recorded challenge response pairs needs to be defined, as this information is sensitive and can be system critical. The latter may be compared to the handling of symmetric device keys, which require a similar level of sensitivity. Besides the initial enrollment, also requirements during the whole lifecycle have to be covered, e.g., the update of keys or helper data, ownership transfer of a device, and taking a device out of service securely.

Based on these points, it becomes even more obvious that a security solution exposing the PUF functionality to other elements needs to designed PUF aware, especially considering reliability and resilience requirements for deployments intended for long usage periods (long term security).

IV. New Applications of Phyiscal Unclonable Functions

The main applications of PUFs fall basically in two categories: A PUF is used directly for object authentication using challenge-response authentication, e.g., for low cost RFID Tags. Here, the PUF is used by the authenticated object to determine the response. Another way of using a PUF is to reconstruct a symmetric cryptographic key that is determined by a fuzzy key extractor using the device's PUF and stored helper data. The reconstructed cryptographic key can be used independently from the specific PUF properties. That means, any cryptographic security mechanism can be used based on the reconstructed key material.

In this section, we describe potential new applications of PUFs in the context of security services.

A. Authentication Verification

It is known to use a PUF to authenticate an integrated circuit or a device respectively. Here, a PUF is used to authenticate the device on which the PUF is realized. However, the reverse usage of a PUF is possible as well: The PUF can be used to *verify* access of an external party. This approach has the clear advantage that no cryptographic algorithm has to be implemented to perform authentication checks. Furthermore, no cryptographic key has been established and stored on the verifier device. An entity that authenticates towards the device has to store a certain number of PUF challenge-response pairs as authentication credentials. The verifying device uses the PUF to check the validity of challenge-response pairs provided by the authenticating entity.

One application for this authentication verification can be, e.g., in the context access verification to a diagnosis or debug interface of an integrated circuit, or to protect the wake-up functionality of a battery-power Internet of Things device or a wireless sensor node.

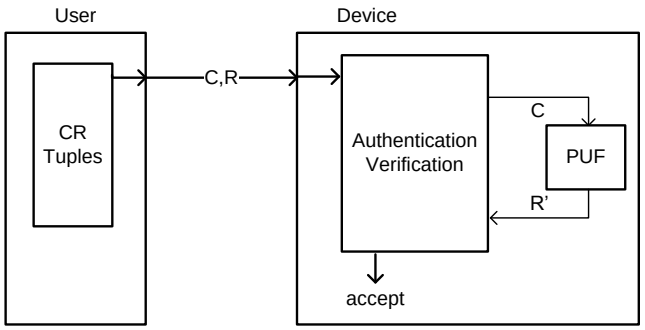

Figure 5. PUF Authentication Verification

Figure 5 illustrates how a PUF can be used to check authenticated access to a device or device functionality by a user as authenticating entity: A user presents a C/R pair of challenge C and response R. The PUF determines the response R' for the given challenge C. If the presented response R, and the determined response R' are identical or differ only in a limited number of bits, access is granted (accept). To increase attack robustness, it may be required to provide multiple valid C/R pairs. The set of challenges of the C/R pairs may be checked for validity. This means that no arbitrary set of CR pairs may be presented, but that the set of challenges of the presented CR pairs is verified to be a valid set of challenge values (e.g., consecutive values). Furthermore, different authorization levels may be associated with different sets of challenge values. So, depending on the presented set of C/R pairs, access to different functionality is granted (e.g., read diagnostic information, or modify configuration parameters).

The C/R pair, respectively the set of C/R pairs, can be determined in different ways:

- During an initialization phase, C/R pairs can be read out from the device, and stored in a secure data base of the authenticating entity. Before the IC is put in operation, the interface allowing to read out C/R pairs is blocked, e.g., by burning a security fuse.
- During an initialization phase, a first set of C/R pairs is read out from the device, and stored in a secure data base. Before the IC is put in operation, the interface allowing to read out C/R pairs without having authenticated is blocked, e.g., by burning a security fuse. Only after having authenticated successfully, the device would allow to read-out additional C/R pairs.
- Should the PUF be a PUF for which an algorithmic model can be determined (as described in [21], the algorithmic model of the PUF can be used to compute C/R pairs. During an initialization phase, C/R pairs are read out to determine the parameters of the PUF model for this device. Before the IC is put in operation, the interface allowing the reading out of C/R pairs can be blocked, e.g., by burning a security fuse.

This inverse usage of a challenge-response PUF for verifying an authentication has the advantage that the verifying device uses the PUF only internally during the operation phase. After the initialization phase, it does not offer an external interface to unauthenticated users that allows to access the PUF directly, i.e., to get responses for given challenges. So, PUF modeling attacks [26] requiring access to PUF responses are avoided.

B. Configuration Integrity Check

In a similar way, the integrity of externally stored configuration data can be verified by a device using its PUF. The configuration data, as the model and serial number, and the configuration and calibration data of a sensor element, can be stored, e.g., in an electrically erasable programmable read only memory (EEPROM).

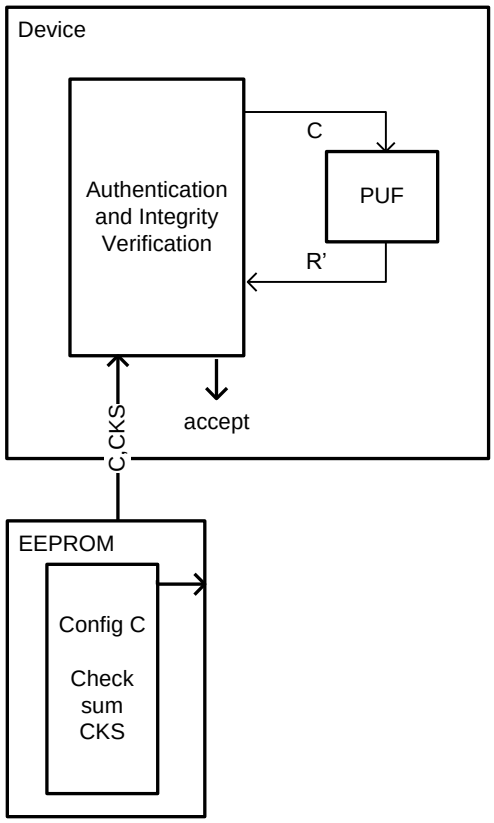

Figure 6. PUF-based Protection of Configuration Data

Figure 6 shows a system where a device uses a PUF to check the integrity of configuration data loaded from an EEPROM memory using its internal PUF. A specific case of configuration data is PUF helper data. The helper data used to reconstruct a cryptographic key could be checked for integrity and authenticity before it is applied with a PUF fuzzy key extractor.

Figure 7 shows the performed steps of a realization option: Configuration data is read from an external, unprotected configuration memory, e.g., a serial EEPROM. Besides the actual configuration data (CD), a PUF checksum (PCS) is also stored on the EEPROM. Once the

configuration CD and its PUF checksum PCS have been read, the device verifies the integrity of the read data using its PUF. A PUF challenge value is determined depending on the read configuration data, e.g., a cryptographic hash value computed over the configuration data. The corresponding PUF response value R' is determined using the device PUF.

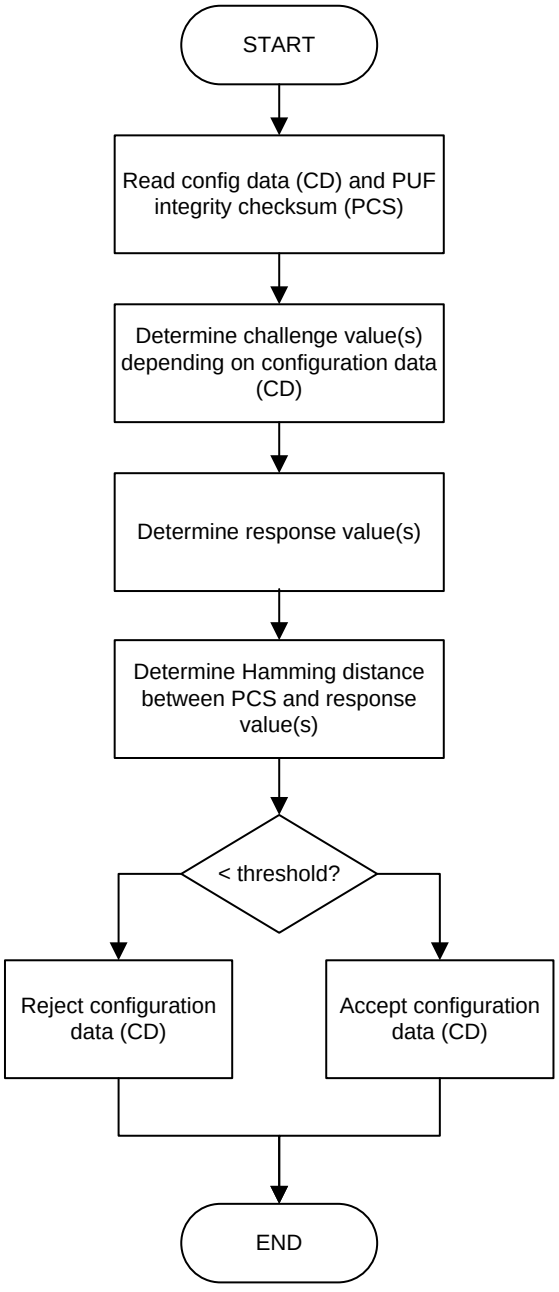

Figure 7. PUF-based Integrity Check of Configuration Data

The Hamming distance, i.e., the number of different bits, between R' and the read PCS value is determined. The read configuration data are accepted if the number of different bits is below a given threshold value.

When writing modified configuration data CD' by the device, the device performs similar steps: The device computes the hash value of the modified configuration data CD' that is to be written to the serial EEPROM. Depending on the hash value, a challenge value is determined.

The device PUF is used to determine the corresponding response as checksum PCS'. The configuration data CD' and the checksum PCS' are written to the serial EEPROM.

This usage of a PUF allows the device to check the integrity of configuration data read from a serial EEPROM. It is ensured that the configuration data has in fact been written by the device. If an attacker should have modified the configuration data on the EEPROM, the checksum would not match the manipulated configuration data. The device would reject the read configuration data, and go into an error state or use internal default values. In this use case, a PUF is used in a similar way as a keyed hash function to determine respectively to check a message authentication code over a given data.

In a similar way as the computation of a message authentication code, a PUF may be used also as part of a cryptographic key derivation function for a cryptographic key *K*. Thereby, a hardware-bound key derivation function is realized. Depending on the cryptographic key *K*, challenge values are determined. The PUF response value(s) are used to determine a (derived) key.

C. PUF Tamper Sensor and PUF Built-In Self Test

Challenge response pairs of the PUF are typically stored as reference data internally within a device. The integrated circuit uses the PUF and the reference data to check whether the PUF circuit is working correctly.

This approach can be used for different purposes:

- A PUF-based tamper-sensor can be realized: The PUF is used as digital sensor to determine a tampering information. When a tampering of the device occurred, the PUF provides different response value with a certain probability. Here, the result of the PUF is not used directly for an authentication or for determining a cryptographic key, but to generate information about the tamper status of a device. If a malfunction is detected, the device can e.g., block a functionality of the device, or it could zeroize stored cryptographic keys.
- A PUF-based device degradation sensor can be realized: Besides physical tampering, also other physical reasons like degradation affect the PUF behavior. A degradation of a device, e.g. a semiconductor IC, can be detected by a changing PUF behavior. The PUF allows detecting such degradations before a regular digital circuit realized on the same semiconductor shows a malfunction. So hardware defects can be detected early, i.e., before the devices shows a failure (predictive maintenance).
- A built-in PUF self-test functionality can be realized. Before a PUF is used, e.g., for authentication or key extraction, its correct operation is verified. Only if the PUF works as expected, the self-test succeeds. The main

function of the PUF, i.e., authentication or key extraction, is performed only when the PUF self-test has succeeded.

Figure 8 shows a realization option where reference data (RD) are used to check the PUF. Only if the PUF provides responses sufficiently similar to the reference data, access to the PUF is enabled by the PUF self-test unit PST. So, an integrated self-test functionality is realized for the PUF. It can be used, e.g., for key extraction or authentication, only if the PUF has passed the self-test successfully.

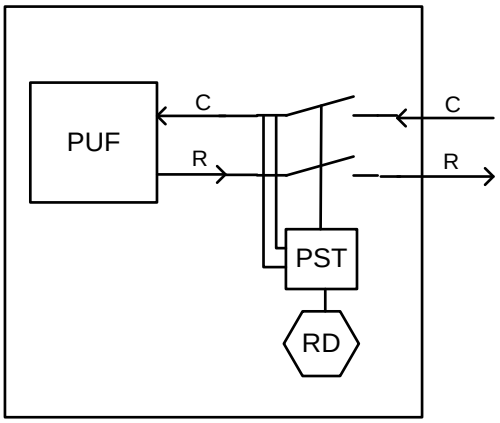

Figure 8. PUF Built-In Self Test

D. Identifying Communication Sender

A PUF can be used to derive a serial number of a device. This PUF derived serial number or a derivation of thereof can be used to determine an identifier for data communication.

For example, an IPv6 stateless address auto configuration can be performed using a PUF. Aura defines how an IPv6 address can be created cryptographically [28]. Similarly, a PUF can be used to determine an IPv6 address. The challenge can be determined based on network part of the IPv6 address assigned by an IPv6 router. The host part is created depending on the PUF response output.

Figure 9 shows a different variant where the PUF-based identifying information is not included in the sender address. Instead, if a wireless spread spectrum transmission system is used, a spreading code is build or modified respectively depending on the PUF response. Hence, the PUF is used to realize a kind of "stream cipher" as spreading code.

A PUF can be used to determine a watermarking information also for digital media, not only for a wireless transmitted information stream. For example, a PUF-based identifying noise signal can be created using a PUF. The noise signal is embedded in a picture, or a video stream as physical watermarking information. Here, a PUF is used to determine the spread spectrum watermarking signal. Instead of a cryptographic signature based watermark as known from [29], a PUF-based identifying watermark is embedded in the digital media. So, a hardware-bound watermarking information can be created, e.g., to prove the originating device that created the media.

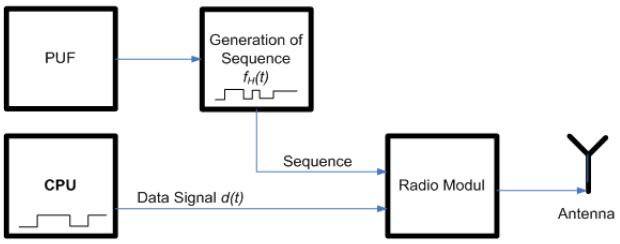

Figure 9. PUF-based Spread-spectrum Transmission

E. PUF-helper Data as License File (license key)

A fuzzy key extractor allows determining a given cryptographic key using a PUF and stored helper data. The helper data has two purposes: it allows correcting random errors of the PUF response, and it transforms the device-specific PUF response to a given cryptographic key. These properties can be used to realize a licensing mechanism, e.g., for feature activation on an embedded device.

In a licensing scheme, a license code, or license key, is required to use a certain, software-based feature. The license code/key can be checked to determine whether a certain feature is allowed to be used, or a cryptographic key to decrypt executable code can be determined based on the license key.

With PUF helper data being required to determine a certain cryptographic key, the license code/key can be provided in the form of helper data: as long as the required helper data is not available, the license key cannot be built by a certain device. However, if helper data to reconstruct a certain license code/key is provided, the device can determine the license code/key. As a PUF is used, the helper data can be processed only on the single intended target device to reconstruct the license information. So, the helper data that is used as license information is valid only on a certain device.

During manufacturing, the executable code for several licensable features would be stored on the device in encrypted form. Without a license, the features cannot be used. To enable the usage of a feature as part of a feature activation procedure, a license code is provided to the device and stored: The license code in the form of helper data enables the device to determine the cryptographic key required to decrypt and execute the code for a certain licensable feature. As different features can be encrypted with a different cryptographic key, the features can be activated independently.

During manufacturing, the PUF of a certain device would be measured. Parameters of the PUF or challenge response pairs would be stored in a device database of the manufacturer. When a certain feature shall be activated for a certain device, the manufacturer uses the stored PUF data to compute offline helper data for the target device. The helper data is constructed in such a way that the resulting cryptographic key is the one needed to decrypt the code for the feature to be activated. So the helper data is not computed by the device itself, but offline by the manufacturer.

F. PUF based Credential Bootstrapping

Cryptographic credentials, as a symmetric or asymmetric cryptographic key, can be used to authenticate a device. The device authentication credentials have to be configured initially by a bootstrapping process (also called enrollment).

Automated credential bootstrapping or enrollment refers to the initial configuration of devices including the key material. This is shown in Figure 10. Field devices are connected to the network, and contact the public key infrastructure (PKI) server to obtain certified key material. Here, the field devices generate their public/private key pairs locally, and send a Certificate Signing Request (CSR) for the public key to the PKI server. Part of the CSR may be a serial number of the device, against which the PKI server can check a configured list of devices allowed to be enrolled. This authorization may also be realized by other means like one-time passwords.

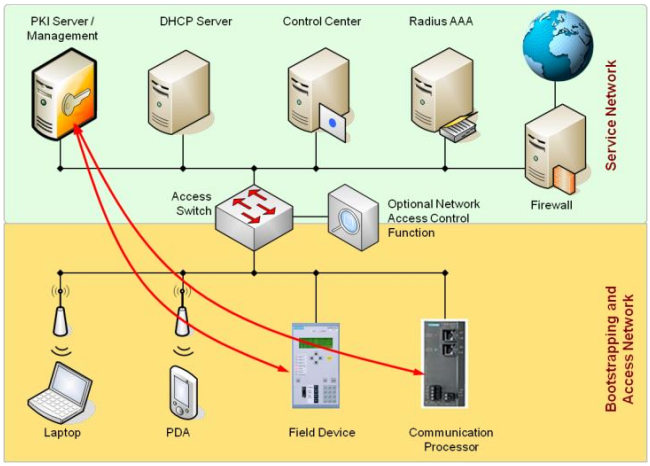

Figure 10. Automated distribution using management protocols

The initial credential bootstrapping is protected usually by organizational and personal security measures. For example, the credential bootstrapping of cryptographic device credentials can be performed during the manufacturing within a secure manufacturing area. Credentials are typically provisioned at the end of a production line. So during the manufacturing and testing of a device, often no credentials are available yet. The manufacturing can be performed at a single manufacturing site, or across different manufacturing sites.

In early manufacturing phases, e.g., directly after the assembly of a digital circuit board, no cryptographic credentials are available yet. However, the device can already be identified using its PUF fingerprint, anyhow. A PUF can be used to identify a manufactured device during the manufacturing process securely along different manufacturing steps *before* cryptographic device credentials have been provisioned. At later steps of the production, or at delivery time to a customer, it can be verified that a certain device is still the same device as intended by checking its PUF fingerprint. So here, a PUF is used not during the regular operation of the device, but during the manufacturing until initial set of cryptographic device credentials have been provisioned.

The provisioning of device certificate using a certificate management protocol like the simple certificate enrolment protocol (SCEP) [30], enrollment over secure transport (EST) [31], or the certificate management protocol CMP [32], a PUF-based device authentication may be used to protect a certificate request message.

In a similar way, a PUF may still be used when cryptographic credentials are not available anymore or are not valid anymore during the lifetime of a device: The validity period of cryptographic credentials may have expired, or credentials may have been zeroized when a device has been decommissioned. A PUF, which is intrinsically available on a device independently on provisioned configuration data and cryptographic credentials, may still be used to identify a device with a certain reliability independently of cryptographic credentials. It can be used as basis to protect the re-provisioning of a device with fresh cryptographic credentials.

V. Related Work

Authentication within the Internet of Things is an active area of research and development. Gupta described multi-factor authentication of users towards IoT devices [33][33]. The Cloud Security Alliance published recommendations on identity and access management within the IoT [34]. Ajit and Sunil describe challenged to IoT security and solution options [35]. Authentication systems for IoT where analyzed by Borgohain, Borgohain, Kumar and Sanyal [36].

Al Ibrahim and Nair have combined multiple PUF elements into a combined system PUF [37].

Host-based intrusion detection systems (HIDS) as SAMHAIN [38] and OSSEC [39], analyze the integrity of hosts and report the results to a backend security monitoring system.

VI. Conclusion

Physical unclonable functions have been investigated extensively by both research, and industry. The work focuses much on design constructions to realize a PUF, analyzing their statistical, and security properties, and on key extraction. Although being known for at least 10 years, one limited number of examples for commercial applications exists. Besides the classical usages, object authentication, and key extraction, a PUF can be specific new usages can be realized based on a PUF. This paper described several new possible applications for PUFs in different systems, either self-contained, like a PUF-based tamper sensor or degradation sensor, or in conjunction with other parts of target solutions like in the case of licensing of credential provisioning. These new applications are discussed as abstract concepts and need to be integrated as security solution element in an overall security solution.

Issues for the practical application of PUFs are the stability over time (ageing), and under harsh environmental conditions. As PUFs are still a relatively new security feature that is not yet broadly applied in practice, careful analysis of

the actual security level as to be performed (e.g., modeling attacks, physical attacks, side channel attacks). A PUF may be used as one security element in an overall security solution design. The security management of a PUF-based security solution has to be designed (e.g., enrollment of key material or helper data, building and maintaining databases comprising challenge/response pairs, update and revocation of security management data along the lifecycle).

However, PUFs show unique properties that make them interesting for practical usage: they allow "storing" a cryptographic key in a protected way without requiring physical non-volatile memory. Low-cost authentication solutions can be built that do not require implementations of cryptographic algorithms. They may be used when conventional cryptographic security mechanisms cannot be applied, or in combination with such security mechanisms. PUFs may be used on low-end devices that do not support cryptographic mechanisms, as additional protection mechanism complementing cryptographic security measures (defense in depth), or during lifecycle phases of a device in which cryptographic credentials are not available. Such lifecycles may occur during production before device credentials have been provisioned or during the lifetime of a device when it is decommissioned or re-provisioned.

REFERENCES

[1] R. Falk and S. Fries, "New Directions in Applying Physical Unclonable Functions," The Ninth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), 23-28 August 2015, Venice, Italy, Thinkmind, pp. 31-36, available from: https://www.thinkmind.org/index.php?view=article&articleid=securware_2015_2_20_30028, last access: January 2016

[2] IEC 62443, "Industrial Automation and Control System Security" (formerly ISA99), available from: http://isa99.isa.org/Documents/Forms/AllItems.aspx, last access: January 2016

[3] B. Gassend, "Physical Random Functions", Masters Thesis, MIT, February, 2003, available from: http://csg.csail.mit.edu/pubs/memos/Memo-458/memo-458.pdf, last access: January 2016

[4] C. Herder, Y. Meng-Day, F. Koushanfar, and S. Devadas, "Physical Unclonable Functions and Applications: A Tutorial," Proceedings of the IEEE, Vol.: 102 No. 8, Aug. 2014, pp. 1126-1141, available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6823677, last access: January 2016

[5] S. Devadas, "Physical Unclonable Functions and Applications, Presentation Slides," available from: http://people.csail.mit.edu/rudolph/Teaching/Lectures/Security/Lecture-Security-PUFs-2.pdf, last access: January 2016

[6] G. E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," DAC 2007, June 4-8, 2007, pp. 9-14 ACM, available from: http://www.verayo.com/pdf/2007_PUF_dac.pdf, last access: January 2016

[7] S. Katzenbeisser, Ü. Kocabas, V. Rožic, A. Sadeghi, I. Verbauwhede, and C. Wachsmann, "PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon," IACR eprint 2012/557, Sep. 2012. Online]. Available from: https://eprint.iacr.org/2012/557.pdf, last access: January 2016

[8] Intrinsic ID Technology, available from: https://www.intrinsic-id.com/technology/, last access: January 2016

[9] Verayo, "Physical Unclonabel Functions (PUF)," available from: http://verayo.com/tech.php, last access: January 2015

[10] Verayo, "Introduction to Verayo," available from: http://www.rfidsecurityalliance.org/docs/Verayo_Introduction_RFIDSA_July_9_08.pdf, last access: January 2016

[11] Microsemi, "SmartFusion2," available from: http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2, last access: January 2016

[12] NXP Semiconductors, "PUF - Physical Unclonable Functions Protecting next-generation Smart Card ICs with SRAM-based PUFs," February 2013, available from: http://www.nxp.com/documents/other/75017366.pdf, last access: January 2016

[13] C. Böhm and M. Hofer, "Physical Unclonable Functions in Theory and Practice," Springer, 2012

[14] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data," Eurocrypt 2004, LNCS 3027, Springer, 2004, pp. 523-540, available from: http://www.iacr.org/archive/eurocrypt2004/30270518/DRS-ec2004-final.pdf, last access: January 2016

[15] B. Škorić, P. Tuyls, and W. Ophey, "Robust key extraction from Physical Uncloneable Functions," Applied Cryptography and Network Security, LNCS 3531, Springer, 2005, pp 407-422, available from: http://members.home.nl/skoric/security/PUF_KeyExtraction.pdf , last access: January 2016

[16] Y. Alkabani and F. Koushanfar, "Active Hardware Metering for Intellectual Property Protection and Security," 16th USENIX Security Symposium, 2007, pp. 20:1-20:16, available from: http://www.usenix.org/event/sec07/tech/full_papers/alkabani/alkabani.pdf, last access: January 2016

[17] M. Gora, A. Maiti, and P. Schaumont, "A Flexible Design Flow for Software IP Binding in FPGA," IEEE Transactions on Industrial Informatics, vol. 6, issue 4, Nov. 2010, pp. 719-728

[18] R. Nithyanand and J. Solis, "Theoretical Analysis: Physical Unclonable Functions and the Software Protection Problem," IEEE Symposium on Security and Privacy Workshop, 2012, pp. 1-11 available from: http://www.ieee-security.org/TC/SPW2012/proceedings/4740a001.pdf, last access: January 2016

[19] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling Attacks on Physical Unclonable Functions," Proc. of the 17th ACM conference on Computer and communications security, 2010, pp. 237-249, , available from: http://people.idsia.ch/~juergen/attack2010puf.pdf, last access: January 2016

[20] K. Rosenfeld, E. Gavas, and R. Karri, "Sensor Physical Unclonable Functions," IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), June 2010, pp. 112-117, available from: http://isis.poly.edu/~kurt/papers/sensorpuf.pdf, last access: January 2016

[21] W. Bares, S. Devadas, V. Khandelwal, Z. Paral, R. Sowell, and T. Zhou, "Soft message signing," patent application, WO2012154409, Nov. 2012

[22] S. Devadas and T. Ziola, "Securely field configurable device," patent application, US2010/0272255, Oct. 2010

[23] M. van Dijk and U. Rührmair, "Physical Unclonable Functions in Cryptographic Protocols: Security Proofs and Impossibility Results," Cryptology ePrint Archive: Report

2012/228, April 2012, available from: https://eprint.iacr.org/2012/228.pdf, last access: January 2016

[24] M. Majzoobi, M. Rostami, F. Koushanfar, D. Wallach, and S. Devadas, "Slender PUF Protocol: A lightweight, robust, and secure authentication by substring matching," IEEE CS Security and Privacy Workshop, 2012, pp. 33-44, available from: http://www.ieee-security.org/TC/SPW2012/proceedings/4740a033.pdf, last access: March 2015

[25] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Side-Channel Analysis of PUFs and Fuzzy Extractors," Conference on Trust and Trustworthy Computing (TRUST 2011), LNCS 6740, Springer, 2011, pp. 33-47

[26] A. Mahmoud, U. Rührmair, M. Majzoobi, and F. Koushanfar, "Combined Modeling and Side Channel Attacks on Strong PUFs," Cryptology ePrint Archive, Report 2013/632, 2013, available from: http://eprint.iacr.org/2013/632, last access: January 2016

[27] C. Helfmeier, C. Boit, D. Nedospasov, S. Tajik, and J.-P. Seifert, "Physical Vulnerabilities of Physically Unclonable Functions," Proceedings of the conference on Design, Automation & Test in Europe (DATE'14), Dresden, Germany, 24-28 March 2014, available from: http://www.date-conference.com/files/proceedings/2014/pdffiles/12.2_5.pdf, last access: January 2016

[28] T. Aura, "Cryptographically Generated Addresses (CGA)," RFC3972, March 2005, available from: https://www.ietf.org/rfc/rfc3972.txt, last access: January 2016

[29] U. Fiore and F. Rossi, "Embedding an Identity-Based Short Signature as a Digital Watermark," Future Internet 2015, 7(4), pp. 393-404, available online: http://www.mdpi.com/1999-5903/7/4/393, last access: January 2016

[30] P. Gutman and M. Pritikin, "Simple Certificate Enrolment Protocol SCEP," draft-gutmann-scep-01.txt, Internet draft, work in progress, September 2015, available from: https://tools.ietf.org/html/draft-gutmann-scep-01, last access: January 2016

[31] M. Pritikin, P. Yee, D. Harkins, "Enrollment over Secure Transport," RFC7030, October 2013, available from: https://www.ietf.org/rfc/rfc7030.txt, last access: January 2016

[32] C. Adams, S. Farrell, T. Kause, T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)," RFC4210, September 2005, available from:, available from: https://www.ietf.org/rfc/rfc4210.txt, last access: January 2016

[33] Udit Gupta, "Application of Multi factor authentication in Internet of Things domain: multi-factor authentication of users towards IoT devices", Cornell university arXiv:1506.03753, 2015, available from: http://arxiv.org/ftp/arxiv/papers/1506/1506.03753.pdf , last access: April 2016

[34] Arlene Mordeno and Brian Russel, "Identity and Access Management for the Internet of Things - Summary Guidance," Cloud Security Alliance, 2015, available from: https://downloads.cloudsecurityalliance.org/assets/research/internet-of-things/identity-and-access-management-for-the-iot.pdf , last access: April 2016

[35] Jha Ajit and M.C. Suni, "Security considerations for Internet of Things," L&T Technology Services, 2014, http://www.lnttechservices.com/media/30090/whitepaper_security-considerations-for-internet-of-things.pdf , last access: April 2016

[36] T. Borgohain, A. Borgohain, U. Kumar, S. Sanyal, "Authentication Systems in Internet of Things," Int. J. Advanced Networking and Applications, vol. 6, issue 4, pp. 2422-2426, 2015, available from http://www.ijana.in/papers/V6I4-11.pdf , last access: April 2016

[37] O. Al Ibrahim and S. Nair, "Cyber-Physical Security Using System-Level PUFs," 7th International Wireless Communications and Mobile Computing Conference (IWCMC), 2011, available from http://lyle.smu.edu/~nair/ftp/research_papers_nair/CyPhy11.pdf , last access: April 2016

[38] R. Wichmann, "The Samhain HIDS," fact sheet, 2011, available from http://la-samhna.de/samhain/samhain_leaf.pdf , last access April 2016

[39] OSSEC, "Open Source HIDS SECurity," web site, 2010 - 2015, available from http://ossec.github.io/ , last access April 2016