Using Physical Unclonable Functions for Internet-of-Thing Security Cameras

Rosario Arjona, Miguel A. Prada-Delgado, Javier Arcenegui, and Iluminada Baturone

Instituto de Microelectrónica de Sevilla (IMSE-CNM)
Universidad de Sevilla, Consejo Superior de Investigaciones Científicas (CSIC)
{arjona,prada,arcenegui,lumi}@imse-cnm.csic.es

Abstract. This paper proposes a low-cost solution to develop IoT security cameras. Integrity and confidentiality of the image data is achieved by using the cryptographic modules that implement symmetric key-based techniques which are usually available in the hardware of the IoT cameras. The novelty of this proposal is that the secret key required is not stored but reconstructed from public data and from the start-up values of a SRAM in the camera hardware acting as a PUF (Physical Unclonable Function), so that the physical authenticity of the camera is also ensured. The variability of the start-up values of the SRAM is also exploited to change the IV (Initialization Vector) in the encryption algorithm, thus increasing security. All the steps to be carried out by the IoT camera at enrollment and normal operation can be included in a simple firmware to be executed by the camera. In addition, this firmware can be trustworthy updated. There is no need to include specific hardware (such as Trusted Platform Modules, TPMs) but only a SRAM is needed which could be powered down and up by firmware.

Keywords: Internet of Things (IoT), Security Cameras, Physical Unclonable Functions (PUFs), Trust and privacy in IoT.

1 Introduction

Many IoT (Internet-of-Thing) security cameras employed nowadays are relatively easy to compromise. Some of them communicate without authentication tokens in plain text (even user credentials, including passwords, are transmitted in plain text). Besides, they accept firmware that is not digitally signed. Hence, they can be exploited for malicious purposes, such as spying or acting as remotely controlled bots to spread malware, for example Mirai, which produced a large disruptive DDoS (Distributed Denial-of-Service) attack at the end of 2016 [1]. In addition, they can suffer man-in-the-middle attacks known as virtual camera attacks, through software tools such as Virtual Webcam, ManyCam or Magic Camera, which are able to modify the image captured by the camera and to simulate captures at real time. This kind of attacks is especially problematic in surveillance or access control applications [2].

Thus, it is becoming increasingly important to include security functionalities into networked cameras: (a) authentication to ensure that the images do not come from a counterfeit camera; (b) integrity of the images to ensure that images have not been altered during communication and they are fresh; and (c) confidentiality to safeguard the privacy of sensitive data, which are subject to legal regulations [3]-[5].

Among the forensic techniques used to authenticate a camera, the simplest one is to evaluate the metadata based on the standard Exif (Exchangeable image file format) [3]. However, the Exif information is stored without protection so that it can be modified or removed. A networked camera can be also identified by its MAC (Media Access Control) address. However, MAC address can be also faked. Other more complicated techniques analyze the peculiarities that each particular image sensor could introduce in its images [6].

Cryptographic techniques are the standard techniques to provide security. There are two types of cryptography: symmetric and asymmetric. The symmetric cryptography techniques employ the same key to cipher and decipher, which should be secret for the sender and the receiver. The problem is the communication of secret keys. The asymmetric techniques employ two types of keys: public and private keys. One public key is associated to one private key and it is very difficult to obtain the private key from the public key. As an advantage, nonrepudiation is achieved and the communication of the private keys is not required. As a disadvantage, asymmetric techniques are computationally more complex, particularly for multimedia data. Hence, asymmetric techniques are used to interchange keys and to create digital signatures, and symmetric techniques are used to cipher the data [3]-[5]. Digital Rights Management (DRM) schemes are widely used to protect video streams on Internet but they employ cryptographic techniques that are computationally intensive for low cost IoT cameras. Hence, new solutions are being provided to integrate security into embedded cameras [5].

In any case, since security lies in the secrecy of the keys, TPMs (Trusted Platform Modules) have been developed which store and process sensitive data in a software protected domain (ARM Trustzone, Texas Instruments M-Shield, etc.) [5]. However, many cameras are deployed in unattended and human-accessible areas and therefore they are vulnerable to physical attacks that are able to read sensitive information from the memories of those modules [7]-[8]. In addition to this, TPMs are specific modules that can be costly for many IoT cameras.

In the latest years, there is a new research line oriented to obfuscate secret keys with the aid of PUFs (Physical Unclonable Functions) [9]. PUFs allow reconstructing secret keys whenever required so that there is no need to store them in non-volatile memories. PUFs exploit variability in the manufacturing process of electronic circuits such as SRAMs, latches, D Flip-Flops, arbiters, ring oscillators, etc. Since the manufacturing process variability is random and particular of each device, PUF responses cannot be predicted and generate unique identifiers for each device. Hence, physical attacks as well as introducing fake cameras in the network are much more difficult because the secret keys are not stored but they are generated by PUFs.

In the literature, there are proposals for the application of PUFs to multimedia security which are based on the responses of image sensors. In [10], the PUF is a measured response of pixels to a defined incident light illuminating the charge-coupled device in a CCD sensor. In [11] PUFs are created by the dark signal non-uniformity of fixed pattern noise in a CMOS image sensor. Recently, PUFs extracted from electronic circuits have been applied to multimedia security. In [12] PUFs are based on the comparison of frequency counters of ring oscillators specifically implemented in the FPGA of a trusted visual sensor node. The solution provided in this paper is more general because it exploits PUFs based on an external SRAM, which is a typical component of any camera, so that no specific circuitry has to be included in the camera. The start-up values from the memory cells of the SRAM are employed as PUF. These values are lost when the memory is powered down and generated on the fly whenever required, without any need of storage of sensitive data. Besides, a trustworthy registration and firmware update can be employed with the proposed approach to make IoT cameras highly secure [13].

The paper is structured as follows. Section II shows how to use the PUFs in low-cost IoT cameras to cipher and authenticate the images captured. Section III summarizes how to obfuscate secret keys with SRAM-based PUFs as well as to generate nonces. Experimental results of a low-cost IoT camera based on a Raspberry Pi 2 model B provided with CMOS SRAM are shown in Section IV. Finally, conclusions are given in Section V.

2 Image Integrity and Confidentiality

Data confidentiality is usually addressed by symmetric-key encryption. Since the amount of data is high in the case of images and image sequences, some approaches resort to encrypt compressed data or to the use of partial or selective encryption of specific regions or objects of interest [4], [14]-[15]. In any case, encryption algorithms should be selected carefully to provide real-time performance with low-cost IoT cameras, which have constrained computing and memory resources. According to how the data are ciphered, ciphers can be classified into block and stream ciphers. Block ciphers process blocks of bits while stream ciphers encrypt bits individually. Among block ciphers, AES (Advanced Encryption Standard) was approved in 2001 as a US federal standard (FIPS PUB 197) and then, it was included in the ISO/IEC 18033-3 standard. Hence, AES is the dominant symmetric-key algorithm in many commercial applications. Particularly, the hardware of many IoT cameras contains an AES encryption module to off-load the CPU from encryption/decryption tasks.

Concerning integrity, Message Authentication Codes (MACs) are usually employed to obtain authentication tags (or cryptographic checksums). MACs can be obtained from block ciphers or from hash functions. The most popular approach in practice is to use a block cipher such as AES in Cipher Block Chaining (CBC) mode [16]. In this mode, the first iteration of the MAC algorithm is computed with the secret key, an Initialization Vector (IV) and the first block of the data to encrypt. The subsequent plaintext data blocks are xor-ed with the previous ciphertext block before they are encrypted. The MAC of the message is the output of the last round. Chaining mode is preferred to encrypt long messages such as images because each ciphertext block produced not only depends on the plaintext block (and the secret key) but also on the preceding blocks. Encrypting the same plain text using the same key produces the same cipher text if each block is encrypted independently. Chaining mode such as CBC is not usually implemented in hardware. In general, a firmware implementation of the block chaining mode uses the AES hardware accelerator.

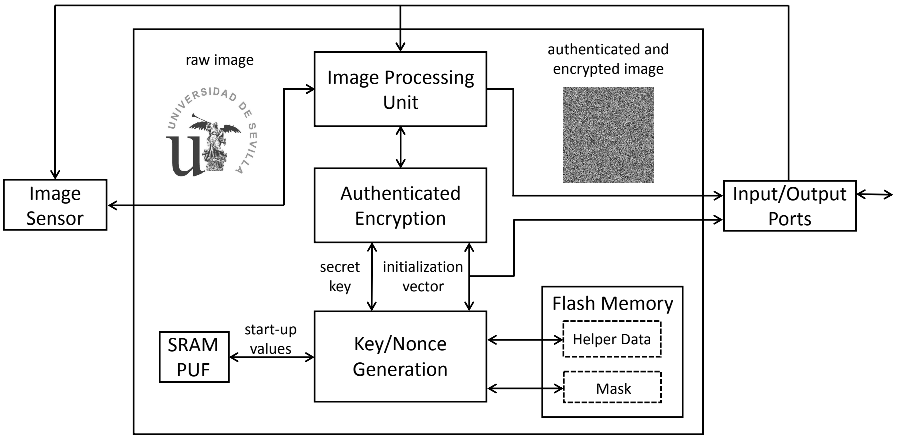


Fig. 1. Authenticated encryption of image data in IoT security cameras.

CBC mode requires a 128-bit IV and the default key size is 128 bits. The IV should never be reused under the same key because it leaks some information about the first block of the plain text. The IV must be also unpredictable at encryption time. If an attacker knows the IV (or the previous block of the cipher text) before the next plain text is specified, the attacker can try to obtain the plain text of some block that was encrypted with the same key before, which is known as the TLS (Transport Layer Security) CBC IV attack.

Several solutions based on chaos theory, cellular automata and DNA computing have also been reported to authenticate encrypted images [17]. However, they do not follow cryptographic standards, so their security can be compromised [18].

We focus on employing standard authenticated encryption algorithms (such as AES-CBC) in IoT cameras, as illustrated in Figure 1. The novelty is that the start-up values of the SRAM included in the IoT camera are employed to obfuscate the secret key of 128 bits and to generate the 128 bits for the IV, as will be explained in the following sections. Whenever the user (or a central server) requests image data from the camera, it power-ups the SRAM PUF and uses non-sensitive data to reconstruct the symmetric secret and to generate a nonce for the IV of the authenticated encryption block. Image and authentication and encryption data are obtained when they are requested and, in this way, the freshness of the image is ensured. The camera communicates the authenticated encrypted image data together with the nonce, so that a genuine receiver is able to decipher the information and verify its integrity with the same authenticated encryption algorithm, using the previously shared secret key and the nonce received. Nonces are used because variable IVs are preferred to increase security.

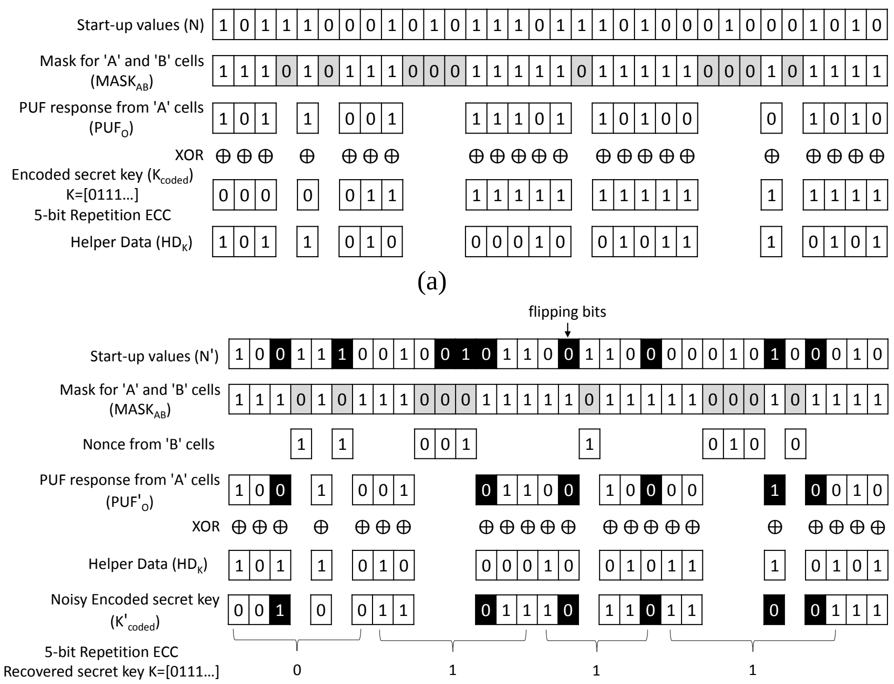
3 Using PUFs to Reconstruct Pre-shared Secret Keys and to Generate Nonces

Since the SRAM cells are composed of two cross-coupled inverters, their start-up values are imposed by the inverter which begins to conduct. The variations of the physical parameters of the transistors make each memory cell have a particular behavior. Such behavior is difficult to predict, to model mathematically, and to clone physically. When the SRAM is powered up, there are memory cells which tend to take the same value ‘0’ or ‘1’ (referred to as stable or ‘A’ cells) and other memory cells which do not take always the same value (referred to as unstable or ‘B’ cells). The start-up values of stable cells are suitable to obfuscate secrets while the start-up values of unstable cells are suitable to generate nonces [19].

The first step to make the secure IoT camera is to register the camera in a controlled environment. The start-up values of the SRAM cells are read several times and the position of ‘A’ and ‘B’ cells, $MASK_{AB}$, which is non-sensitive information, is saved in the device or in the central server. Details about the classification procedure can be seen in [19]. The helper data HD_K, also non-sensitive information, associated to a secret key K, are generated and also saved in the device or the central server. The procedure to generate helper data using code-offset techniques is described in [20]. A pre-shared secret key K is encoded using a linear Error Correction Code (ECC), such as a bit repetition code. The result is K_{coded}= ECC(K). The XOR operation is used to obtain HD_K = XOR[PUF_O, K_{coded}], where PUF_O are the start-up values of a set of stable cells. The ECC is needed because stable cells can show bit flipping, although with low probability. The helper data do not reveal information about the secret key if the bit string PUF_O is random. A required condition for randomness is that the number of 1’s and 0’s in the start-up values should be balanced. Otherwise, a debiasing algorithm should be applied (for instance, von Neumann algorithm) [20].

Once registered, the IoT camera can operate securely in a normal mode. The public data $MASK_{AB}$ and HD_K associated to the camera are employed to reconstruct the shared key K. The camera powers-up the SRAM, selects the same ‘A’ cells used in the registration phase with the $MASK_{AB}$ and obtains a bit string PUF'_O, which will be very similar to PUF_O. It carries out XOR [PUF'_O, HD_K] = K'_{coded}, where K'_{coded} is very similar to K_{coded} so that the last one is recovered (and, hence, the shared secret key) after applying the decoder of the ECC to K'_{coded}. The camera also selects the B cells with the $MASK_{AB}$ and obtains the nonce IV of the authenticated encryption block. The genuine camera communicates the authenticated encrypted image data together with the nonce, so that the genuine receiver is able to decipher the information and verify the integrity of the data and the authenticity of the camera because no other camera is able to reconstruct the shared key even with the public data provided by the server. Figure 2 (a) and (b) illustrate, respectively, the registration and normal mode phases. In the example, an ECC based on the repetition of 5 bits is employed.

The secret key shared between the camera and the receiver should change to increase security. Hence, after several times, a new secret key should be derived from the former one using a Key Derivation Function (KDF) that follows NIST recommendations. Details of this procedure as well as details of how to update the firmware of the camera in a trustworthy way can be seen in [13].



(a)

(b)

Fig. 2. (a) SRAM PUF registration. (b) Secret key reconstruction and nonce generation from SRAM PUF (a 5-bit repetition ECC is used in the example).

The quality of a PUF response is evaluated with two figures of merit [19]. One of them is reliability, that is, two responses from the same PUF (in this case the start-up values of the same SRAM cells) should be very similar. The other one is uniqueness, that is, two responses from different PUFs should be very different. The responses from PUFs can be compared through the Hamming Distance (HD). IntraHD values are obtained by comparing the responses from the same SRAM and InterHD values are obtained by comparing the responses from different SRAMs. Ideally, the IntraHD values should tend to zero (that is, start-up values from several power-ups of the same SRAM are reliable) and the InterHD values should be around 0.5 (that is, the responses are unique of each device and have no bias). Thus, the distribution of the IntraHD values is well separated from the distribution of the InterHD values.

4 Experimental Results of the Proof of Concept

As a proof of concept, a low-cost IoT camera based on Raspberry Pi 2 model B was considered. It employs a Raspberry Pi Camera Module v2 which has an 8-megapixel sensor and can provide video as well as still photographs. The AS6C1008 CMOS

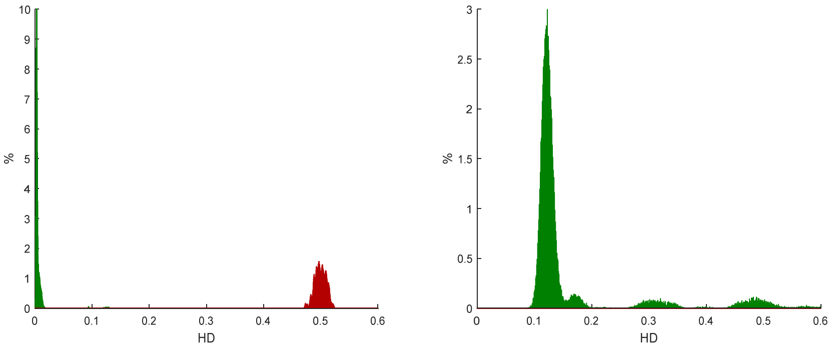
Fig. 3. The IoT camera (on the left), the authenticated encrypted image (in the center), and the original image (on the right).

SRAM which can be connected to the GPIOs of the Raspberry was employed as SRAM PUF. The Picamera library for Python was used to work with the camera module and the RPi.GPIO library allows working easily with the GPIO and, hence, the SRAM PUF. The PyCrypto library was used to manage the AES encryption algorithm and the Tkinter library was employed to program a Graphical User Interface to visualize the results. Figure 3 shows a photograph of the IoT camera and the processing result on a still photograph.

In order to evaluate the PUF responses that can be generated by the SRAM PUF, the analysis was done with measurements of the first 512 words from 20 different SRAMs (each SRAM is organized as 131,072 words by 8 bits). The start-up values from each SRAM were measured 100 times. The first 30 measurements were employed to classify the cells into ‘A’ and ‘B’ cells (thus generating the masks) and the rest of the measurements were employed to evaluate the PUF performance. After obtaining the ‘A’ and ‘B’ cells, the associated sequences were truncated to 2,537 and 1,126 bits, respectively, in order to generate sequences with the same length in all the SRAMs (those sizes were fixed by the smallest masks).

The PUF reliability was evaluated by computing the Hamming Distance between sequences composed of 2,537 start-up values of ‘A’ cells from the same SRAM. For 20 SRAMs and 70 sequences for each one, the total number of comparisons is 48,300 (20x70x69/2). The mean of the Hamming Distance values is 0.51%, which is a value close to the ideal value of 0 %. For the evaluation of the PUF uniqueness, all the sequences of one SRAM were compared to all the sequences of the rest of the SRAMs. For 20 SRAMs and 70 sequences obtained for each one, the total number of comparisons is 931,000 (70x70x20x19/2). In this case, most of the comparisons are located around 49.95%, which is close to the ideal value of 50%. Thus, the sequences have the same number of 1’s and 0’s and there is no bias. These results are illustrated through the IntraHD and InterHD distributions in Figure 4 (a). The SRAMs considered are suitable for obfuscating secret keys: the PUF responses of different SRAMs are quite different so that the helper data do not reveal information about the secret key and only the camera with the genuine SRAM is able to reconstruct the secret key.

A similar evaluation was performed for the generation of nonces from the SRAMs. The nonces were obtained from sequences composed of 1,126 start-up values of ‘B’ cells. The comparisons were 48,300 for the intraHD. The resulting distribution is shown in Figure 4 (b). Most of the intraHD values are around 12%. However, there are intraHD values distributed from 8.61% (minimum intraHD value) to 70.69%



(a) (b)

Fig. 4. (a) IntraHD (on the left) and InterHD (on the right) distributions for sequences composed of ‘A’-cell start-up values. (b) IntraHD distributions for sequences composed of ‘B’-cell start-up values.

(maximum intraHD value). Therefore, sequences obtained from successive start-up values of ‘B’ cells of the same SRAM are sufficiently different to generate nonces.

The schemes for the enrollment and reconstruction of the secret key (described in Section 3) can be implemented as firmware in the IoT camera. The only hardware requirement is that the camera has an SRAM that could be powered down (which is usual to reduce power consumption). At the enrollment phase, the $\mathrm{MASK}_{\mathrm{AB}}$ is created and saved, the camera user (or central server) establishes the master secret key to be used in the communication in a controlled environment (previously, for instance, to place the camera in its final location) and the helper data are created and saved. To cope with the maximum intraHD value measured for the ‘A’ cells (13.32 %), the codeword ($\mathrm{K}_{\mathrm{coded}}$) is generated from the secret key by a 25-bit Repetition Error Correction Code. If the intraHD distribution is modeled as a binomial cumulative distribution and 13.32% is taken as the probability of bit flipping, the probability of a string of 25 bits with more than 12 flipping bits (the probability of reconstructing the key with an error in a bit) is as low as $4.5 \cdot 10^{-6}$. Since secret keys of 128 bits are considered as commented in Section 3, 3,200 start-up values from ‘A’ cells are needed. Since the minimum percentage of ‘A’ cells measured in the SRAMs was 56.91%, 703 words are read from the SRAM. The start-up values from ‘A’ cells are selected from the 703 words using the $\mathrm{MASK}_{\mathrm{AB}}$ created and they are truncated to 3,200 bits to form the PUF response ($\mathrm{PUF}_{\mathrm{O}}$). The codeword ($\mathrm{K}_{\mathrm{coded}}$) is combined with the PUF response ($\mathrm{PUF}_{\mathrm{O}}$) by a XOR operation to result the Helper Data (HD_{K}). $\mathrm{MASK}_{\mathrm{AB}}$ and the HD_{K} are the only data stored. In the normal mode of operation, the 703 words are read again from the SRAM and processed to reconstruct the 128-bit secret key required by the integrated authenticated encryption algorithm, using the $\mathrm{MASK}_{\mathrm{AB}}$ and the HD_{K} stored. The $\mathrm{MASK}_{\mathrm{AB}}$ is also employed to select the start-up values from ‘B’ cells (128 bits) to obtain the nonce/IV required by the AES-CBC encryption algorithm.

Table 1 shows a comparison with other proposals in the literature also aimed at including PUF-based security into cameras [11]-[12]. Our proposal is preferred whenever the hardware of the camera cannot be designed to integrate the PUF into the image sensor [11] or into the controller of the virtual sensor node [12].

Table 1. Comparison with other proposals.

Proposal	Average Intra HD	Average Inter HD	No. devices	Specific hardware
Image sensor PUF [11]	0.20%	49.37%	5	Required
Trusted visual sensor node [12]	1.40%	49.0%	9	Required
This work	0.51%	49.95%	20	Not required

5 Conclusions

This work proposes a low-cost security solution to provide authentication, integrity and confidentiality to the image data obtained from IoT cameras. The solution is based on the use of standard cryptographic modules usually available in the hardware of the cameras (AES module). The 128 bits required for the secret key and the 128 bits needed for the nonce/IV of AES-CBC are generated from the start-up values of a SRAM of the camera acting as a PUF. Experimental results of the intraHD and interHD distributions prove that the PUF responses of standard off-the-shelf SRAMs satisfy reliability and uniqueness requirements for the secret keys and variability requirements for the nonces/IVs. A simple repetition Error Correction Code is enough to correct the low bit flipping of the memory. An IoT camera with a SRAM that could be powered down and up can be registered and updated with a trustworthy firmware that implements the proposed solution. Since SRAMs are available in many sensors and actuators, the proposed solution can be applied to ensure the authentication, integrity and confidentiality of the data generated in many other IoT scenarios.

Acknowledgments. This work was supported in part by TEC2014-57971-R project from Ministerio de Economía y Competitividad of the Spanish Government (with support from the PO FEDER-FSE) and 201750E010 (HW-SEEDS) project from CSIC. The work of Miguel A. Prada-Delgado was supported by V Plan Propio de Investigación through the University of Seville, Seville, Spain.

References

1. Kolias, C., Kambourakis, G., Stavrou, A., Voas, J.: DDoS in the IoT: Mirai and Other Botnets. IEEE Computer, vol. 50, no. 7, pp. 80–84 (2017).
2. Chen, S., Pande, A., Mohapatra, P.: Sensor-Assisted Facial Recognition: An Enhanced Biometric Authentication System for Smartphones. In: Proceedings of the 12th Annual International Conference on Mobile Systems, Applications and Services, pp. 109–122 (2014).
3. Li, C.-T.: Multimedia Forensics and Security. Hershey: Information Science Reference (2009).

4. Lian, S.: Multimedia Content Encryption: Techniques and Applications. CRC Press (2008).
5. Winkler, T., Rinner, B.: Privacy and Security in Video Surveillance. In: Intelligent Multimedia Surveillance: Current Trends and Research, pp. 37–66, Springer (2013).
6. Lukas, J., Fridrich, J., Goljan, M.: Digital Camera Identification from Sensor Pattern Noise. IEEE Transactions on Information Forensics and Security, vol. 1, no. 2, pp. 205–214 (2006).
7. Sanlyde, D., Skorobogatov, S., Anderson, R., Quisquater, J.-J.: On a New Way to Read Data from Memory. In: Proceedings of the First International IEEE Security in Storage Workshop, pp. 65–69 (2002).
8. Choi, P., Kim, D. K.: Design of Security Enhanced TPM Chip against Invasive Physical Attacks. In: IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1787–1790 (2012).
9. Herder, C., Yu, M.-D., Koushanfar, F., Devadas, S.: Physical Unclonable Functions and Applications: A Tutorial. In: Proceedings of the IEEE, vol. 102, no. 8, pp. 1126–1141 (2014).
10. Shokrollahi, J., Martin, C.: Method for Authenticating a Charge-Coupled Device (CCD). Patent No. 8817123 (2014).
11. Cao, Y., Zhang, L., Zalivaka, S. S., Chang, C.-H., Chen, S.: CMOS Image Sensor Based Physical Unclonable Function for Coherent Sensor-Level Authentication. IEEE Transactions on Circuits and Systems-I: Regular Papers, vol. 62, no. 11, pp. 2629–2640 (2015).
12. Haider, I., Höberl, M. Rinner, B.: Trusted Sensors for Participatory Sensing and IoT Applications based on Physically Unclonable Functions. In: Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security (IoTPTS), pp. 14–21 (2016).
13. Prada-Delgado, M. A., Vázquez-Reyes, A., Baturone, I.: Trustworthy Firmware Update for Internet-of-Thing Devices Using Physical Unclonable Functions. In: Global IoT Summit, (2017).
14. Massoudi, A., Lefebvre, F., De Vleeschouwer, C., Macq, B., Quisquater, J.-J.: Overview on Selective Encryption of Image and Video: Challenges and Perspectives. EURASIP Journal on Information Security, pp. 1–18 (2008).
15. Jeon, Y., Kim, Y., Kim, J.: Implementation of a Video Streaming Security System for Smart Device. In: IEEE International Conference on Consumer Electronics (ICCE), pp. 97–100 (2014).
16. National Institute of Standards and Technology. Recommendations for Block Cipher Modes of Operation. NIST special publication 800-38A (2001).
17. Souyah, A., Faraoun, K.M.: A Review on Different Image Encryption Approaches. In: Chikhi, S., Amine, A., Chaoui, A., Kholladi, M., Saidouni, D. (eds) Modelling and Implementation of Complex Systems. Lecture Notes in Networks and Systems, vol 1. Springer (2016).
18. Caragata, D., Mucarquer, J.A., Koscina, M., El Assad, S.: Cryptanalysis of an Improved Fragile Watermarking Scheme. AEU - International Journal of Electronics and Communications, vol. 70, no. 6, pp. 777–785 (2016).
19. Baturone, I., Prada-Delgado, M. A., Eiroa, S.: Improved Generation of Identifiers, Secret Keys, and Random Numbers from SRAMs. IEEE Transactions on Information Forensics and Security, vol. 10, no. 12, pp. 2653–2668 (2015).
20. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. SIAM Journal on Computing, vol. 38, no. 1, pp. 97–139 (2008).
21. Prada-Delgado, M. A., Vázquez-Reyes, A., Baturone, I.: Physical Unclonable Keys for Smart Lock Systems using Bluetooth Low Energy. In: 42nd Annual Conference of the IEEE Industrial Electronics Society (IECON), pp. 4808–4813 (2016).