Smart Cameras with onboard Signcryption for Securing IoT Applications

Subhan Ullah*† Bernhard Rinner* and Lucio Marcenaro†
*Institute of Networked and Embedded Systems
Alpen-Adria-Universität Klagenfurt, Universitätsstraße 65-67, 9020 Klagenfurt, Austria
Email: {subhan.ullah, bernhard.rinner}@aau.at
†Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture
University of Genova, Via all'Opera Pia 11, 16145 Genova, Italy
Email: lucio.marcenaro@unige.it

***Abstract*—Cameras are expected to become key sensor devices for various internet of things (IoT) applications. Since cameras often capture highly sensitive information, security is a major concern. Our approach towards data security for smart cameras is rooted on protecting the captured images by signcryption based on elliptic curve cryptography (ECC). Signcryption achieves resource-efficiency by performing data signing and encryption in a single step. By running the signcryption on the sensing unit, we can relax some security assumptions for the camera host unit which typically runs a complex software stack. We introduce our system architecture motivated by a typical case study for camera-based IoT applications, evaluate security properties and present performance results of an ARM-based implementation.**

***Keywords*-Smart camera; Signcryption; Security; Internet of Things;**

I. Introduction

Smart cameras are real-time video acquisition and processing systems that combine onboard sensing, processing and communication capabilities and play an important role in several IoT applications [1]. However, security and privacy protection has become a major concern due to their widespread deployment, the sensitive nature of the captured data and the open infrastructure [2], [3]. Basic security objectives for a smart camera are thus (i) to prove the originality of images or video data (integrity), (ii) its origin (authenticity of visual sensor) and (iii) to avoid third parties unauthorized access (confidentiality) throughout the entire lifetime of the data.

This paper introduces a security approach for smart cameras by integrating signcryption [4] with elliptic curve (EC) for improving resource efficiency. We have extended the preliminary work on securing the camera node [5] by separating the platform into a trusted sensing unit with exclusive access to the image data and an untrusted camera host unit running user specific applications, operating system, middleware and networking tasks. Such separation helps to mitigate the increasing attack threats for complex embedded software systems [6]. Integrity, authenticity and confidentiality are typically achieved by digital signature and (public key) encryption. Traditionally, these security functions are realized as sequential steps in a *sign-then-encrypt* fashion (e.g. [7]). Signcryption is a resource-efficient technique which implements signature and encryption in a single step and achieves a lower computational and communication cost than the traditional approach [4], [8]. In our approach we apply EC-based signcryption directly on the sensing unit in order to push protection as close as possible to the sensor. The particular challenges for our approach are the resource limitations, the processing of high volume of image or video data, the open infrastructure (e.g., Internet) and the need for real-time performance in IoT.

The contribution of this work lies in the deployment and evaluation of EC-based signcryption directly on the sensing unit. We propose the overall system architecture which is motivated by a smart home surveillance case study and briefly analyze security properties of our approach. The case study is defined as event-triggered monitoring where smart cameras perform onboard event detection and initiate the transfer of protected data to mobile devices and some backup server. We further present runtime measurements on an ARM-based implementation.

The rest of the paper is organized as follows: Section II discusses the state-of-the-art. Section III introduces the system architecture and threat model. Section IV and V describe the proposed solution, experimental setup and results, respectively. Finally, section VI concludes the paper.

II. State-of-the-art

In the following, we briefly discuss selected security techniques for image data. Digital watermarking [9], [10] is a widely used approach for the integrity verification of image data, i.e., to detect any changes in the size or pixel values of images. Schneider and Chang [11] presented a content-based digital signature method to authenticate images and videos. They first extracted the interesting contents from the image, hashed it and then used the private key for generating the signature. Atrey et al. [12] also applied a digital signature scheme to detect spatial cropping and temporal jittering in a video stream. They used three hierarchical levels of videos and converted the input video into shots which were then converted to frames. For each level a signature

978-1-5090-5873-0/17/$31.00

is generated; a master signature is then derived from the individual level signatures using a master key. Mohanty [13] presented a scheme called cryptmark which is based on digital watermarking and advanced encryption standard (AES) encryption techniques for the security of smart cameras as part of an integrated real-time digital rights management (RDM) system. He used a custom designed embedded smart camera prototype based on a field programmable gate array (FPGA) and achieved integrity, authenticity and guaranteed ownership rights for videos.

Another research area is to apply security techniques close to the visual sensor. Nelson et al. [14] proposed a CMOS active pixel sensor (APS) imager with sensor-specific on-chip watermarking. This built-in watermarking was intended towards a pervasive image authentication. Stifter et al. [15] used an on-chip cryptographic unit to secure the image and video data. They achieved the authentication, integrity and freshness of a complete image frame by calculating a checksum derived from message authentication code (MAC). They also equipped the image sensor with a dedicated EEPROM to uniquely identify the imager. Serpanos and Papalambrou [16] suggested that, the image sensor should be trusted to prevent the insertion of unauthorized nodes in a distributed smart camera system. Winkler and Rinner [17] presented a novel platform TrustEYE.M4 using a hardware based trusted platform module (TPM) security chip for onboard security and privacy protection. In [7] they extended their work based on a sign-then-encrypt approach and presented a solution for the secure use of public cloud storage for data archiving and delivering. By using RSA digital signature and time-stamping techniques, they were able to prove non-repudiation and authentication for the captured data. Cao et al. [18] proposed a CMOS image sensor based on physical unclonable function (PUF) for on-chip authentication and identification. They generated a unique and reliable signature by exploiting the dark signal noise uniformity of fixed pattern noise in the CMOS image sensor.

Our proposed signcryption technique implements elliptic curve based digital signature algorithm (ECDSA) and AES in a single step, which provides integrity, authenticity and confidentiality simultaneously for image or video data. The smaller key size of EC [19] and the implementation of signature and encryption in a single step [20] supports real-time data security directly on the sensing unit. To the best of our knowledge, our approach is the first deployment of signcryption in this context.

III. System architecture

The external and internal view of the proposed system architecture is shown in Figure 1.

A. External view

The external view shows the integral components of typical IoT applications. In particular, we envision event-triggered monitoring in a smart home as use case. The primary goals of this use case are: (i) monitoring of specific regions of interests with smart cameras, (ii) onboard detection of predefined events, (iii) transmission of real-time event messages and images to mobile devices, and (iv) storing the collected information on the server. The server also notifies the mobile device when a new image or video data is added to it. The stored information on the server can be easily accessed by the mobile device. Figure 1 highlights the communication between these IoT devices in different colors, e.g., green represents the exchange of alerts or request messages and brown represents the exchange of video frames or images.

B. Internal view

The internal view shows the functionalities of the system, where the sensing unit captures the image sequence of target scene and detects regions of interest (RoI) through predefined low-level video processing algorithms. The RoIs serve as event data and are transfered to the camera host which is responsible for further processing and for transmitting them to the backup server and alerting the mobile device about the detected events.

C. Threat model

In this work, a threat model addresses eavesdropping, data modification, impersonation and replay attacks to the information transmitted from the sensing unit to the mobile device.

1) Assumptions: We assume that the sensing unit is trusted and hence that the attacker has no access to data on the unit. The protection of the sensing unit is build upon our previous work [7] and [21]. The attacker can access the data possibly on the camera host part, communication channels or backup server, as reported in the context of IoT smart home [22] and visual sensor network (VSN) [2] scenarios. The denial of service (DoS) attacks on the camera host or backup server are not explicitly considered in this work. Moreover, we assume that the mobile device is trustworthy and the private keys are securely stored on it. The corruption of software or hardware and DoS attacks on the camera host, backup server or mobile device can disrupt the normal functionality of the proposed system architecture.

2) Eavesdropping attack: Eavesdropping is a passive attack and its goal is to compromise the confidentiality of information during transmission on the communication channel or in any other part of the system e.g., on the camera host or the backup server. Eavesdropping attacks are possible if security credentials such as encryption keys are compromised by the attacker.

3) Data modification: In the case of the data modification attack, the attacker can change, inject or delete information stored on the camera host as well as during the transmission to the mobile device. Compromised keys are common reasons for this attack.

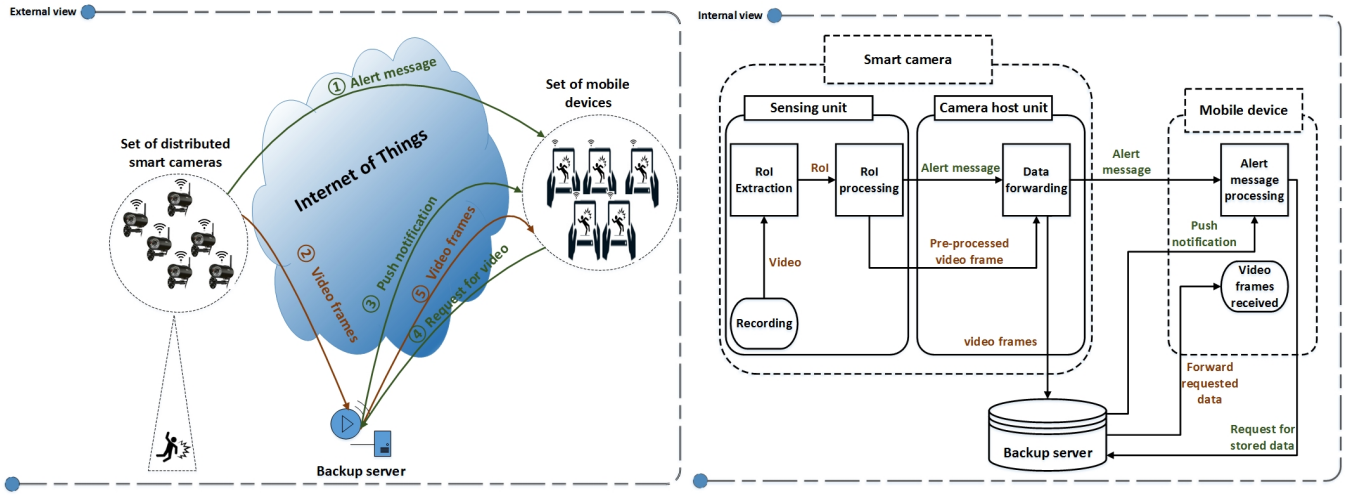

Figure 1. System architecture: (a) The external view consists of a set of distributed smart cameras, a backup server and set of mobile devices communicating with each other. (b) The internal view shows the functionalities and flow of data in the system, where the smart camera is further divided into a sensing unit and a camera host unit. The camera host consists of different hardware and software stacks e.g., operating system, network stack, system libraries, to run and manage the camera application.

4) Impersonation: During an impersonation attack, an attacker successfully uses the identity of a legitimate component to transmit its own data.

5) Replay attack: In a replay attack the same valid information are transmitted by an attacker repeatedly or he modifies timestamps and delivers outdated information as fresh one.

IV. Proposed Solution

In the system architecture shown in Figure 1, the sensing unit extracts the ROIs and generates alert messages from the captured video. This work proposes the signcryption technique to implement signature and encryption in a single logical step directly on the sensing unit to protect the event data (ROIs and alert messages). The sensing unit then transmits the protected event data to the camera host which verifies the received data and forwards the signcrypted alert message and related video frames to the mobile device and the backup server, respectively. When the server receives new data from the smart camera, it sends a push notification to the mobile device. As soon as the mobile device receives the alert message from the smart camera and a push notification from the server, it sends a request to the server to access the required data. Due to the limited storage on the smart camera data is only stored on the backup server. Thus, the data will only be available for further access on the backup server. This approach minimizes the incoming requests on smart camera and allows specific (known requests) only, which on the other hand minimizes the DoS attacks. But the explicit protection of DoS attacks are beyond the scope of this paper.

A. Signcryption

The signcryption technique simultaneously fulfills both the functions of digital signature and public key encryption logically in a single step and provides authentication, integrity, and confidentiality.

1) Signcryption setup: In this work we implement signcryption [20] with ECDSA and public key encryption AES. An one-way keyed hash function and 256 bits AES key are required for signcryption. The setup of the implementation is based on the EC domain parameters [23]. An EC over the finite field F_p is represented by $E(F_p)$ with a base point $G \in F_p$ of order q, where G is chosen randomly from set of points on $E(F_p)$. The parameter p is a prime number specifying the finite field F_p.

2) Keys generation for signcryption: Assume that the private key generator (PKG) generates the private key P_r and the public key P_u using EC for the sensing unit and other devices in our system architecture. The private key is randomly chosen from a set of large prime numbers. The public keys are derived also from the point on the elliptic curve on the basis of the chosen private key, e.g., $P_u = P_r.G$, is called elliptic curve discrete logarithm problem (ECDLP). PKG generates the sensor's private key Pr_{sensor} and the public key Pu_{sensor}. It also generates the mobile device private key Pr_{mobile} and the public key Pu_{mobile}. These keys are distributed by a key distribution center (KDC) in a secure way during initialization of the system or joining of new device.

3) Signcryption algorithm: After distribution of the keys by the KDC, each device securely stores its private key and shares its public key with each other. The sensing unit applies signcryption on the captured video or image frames.

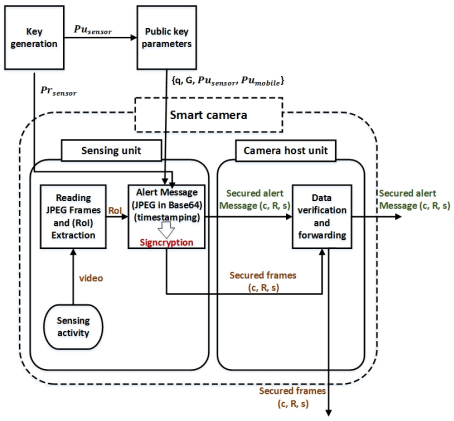

Figure 2. Signcryption model.

The signcryption algorithm chooses a prime number v where $v \in \{1, 2, 3, ...q-1\}$. The signcrypted message and frames represented by green and brown color in the form of (c, R, s) are transferred to the camera host as shown in the Figure 2.

$$k_1 = hash(v.G) \tag{1}$$
$$k_2 = hash(v.Pu_{mobile}) \tag{2}$$
$$c = Enc_{k_2}(frame) \tag{3}$$
$$r = hash(c, k_1) \tag{4}$$
$$s = \frac{v}{(r + Pr_{sensor})} \, mod \, q \tag{5}$$
$$R = (r.G) \tag{6}$$
$$Signcryption \; output = (c, R, s) \tag{7}$$

Then the camera host verifies the authenticity of the signcrypted data with the public key of the sensor and considers it as authentic if $r.G = R$, otherwise the host discards it. By using this property of signcryption, the camera host can verify the authenticity of the data without compromising its confidentiality. After the succesful verification the camera host forwards the secured alert message and frames to the mobile device and the backup server, respectively.

4) Un-signcryption algorithm: When the mobile device receives an alert message and encrypted video frames, it performs the following un-signcryption algorithm as shown in Figure 3.

$$k_1 = hash(s(R + Pu_{sensor})) \tag{8}$$
$$r = hash(c, k_1) \tag{9}$$
$$k_2 = hash(Pr_{mobile}(s(R + Pu_{sensor}))) \tag{10}$$
$$frame = Dec_{k_2}(c) \tag{11}$$
$$r.G = R \tag{12}$$

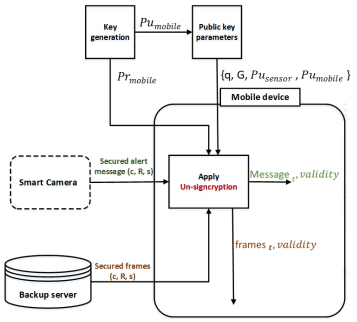

Figure 3. Un-signcryption model.

B. Security analysis

A security analysis of the signcryption scheme (section IV) with specific attention to the system architecture (section III) is presented, in order to countermeasure the attacks identified in our threat model (section III-C). The basic security goals of these countermeasures are confidentiality, integrity, authenticity, freshness of the data processed by the sensing unit. The security of signcryption is based on the assumption of computational hardness of ECDLP [24].

1) Confidentiality: Confidentiality of image or video frames is provided by AES encryption using a session key k_2 during the signcryption process. k_2 is derived by using a secret key v and the public key of the mobile device Pu_{mobile} (cp. Equ. (2)). In this case, the attacker needs to know v to derive k_2. To guess v corresponds to solving the ECDLP. Another possibility for an attacker is to solve Equ. (10), but in this case the attacker only knows the public key of the mobile device Pu_{mobile} but not the private key of mobile device Pr_{mobile}. To derive the private key of the mobile device attacker needs to solve the ECDLP again. It means that the encryption key is secure from both sensing unit and mobile device perspective to the attacker.

2) Integrity: The sensing unit processes a valid signcryption part r by hashing the encrypted data c with k_1 as shown in Equ. (4). In this case anyone can check the integrity of the encrypted data using k_1 derived from the associated public key of sensor, s and R as shown in Equ. (8). If an attacker modifies the encrypted data c to c', the change will be detected on the mobile device because of collision resistance of the hash function. This technique will provide the integrity of the single image or of video frame data as well as the correct ordering of all the frames.

3) Authentication: In the proposed system model, it is important to know the identity of the sensor of the smart camera which is claiming the capturing of the image or video data. The signcryption technique provides the authentication by using the following proofs e.g.,
if $hash(s(R + Pu_{sensor})) = hash(v.G) = (k_1)$ from Equs. (1) and (8).

4) Freshness of the captured data: Timestamping provides freshness of data and prevents replay attacks. We assume that image or video frames are securely timestamped before signcryption. The mobile device verifies the validity and timestamp by un-signcryption of the image.

V. Experimental setup and results

The implementation of the proposed EC-based signcryption is performed on a Raspberry Pi-3 which has an 1.2 GHz ARMv8 CPU and 1 GB RAM. A Pi-camera sensor is used to capture images in JPEG format. The images are stored in Base-64 encoding to enable AES encryption during the signcryption process. Java is used for implementation because of its portability, its built-in security features, and the open source Java libraries for EC computation. To evaluate the efficiency of signcryption technique, we integrated signcryption and unsigncryption in a single Java package and measured the running times. We investigated the running times for protecting single images in two different experiments. In the first experiment (as shown in Fig. 4), we varied the key size for EC (192, 256 and 384 bits) and kept the image size fixed to 105 kB. In the second experiment (as shown in Fig. 5), we varied the image size (68, 105 and 180 kB) and kept the key size fixed to 384 bits. The results show that the running time is only slightly influenced by these variations. Although the image size is almost tripled, the running time only varies by 5 % for signcryption and 11 % for unsigncryption, respectively. The computationally expensive part of signcryption and unsigncryption are EC-point operations. Signcryption has a slightly longer running time because it requires three EC-point multiplications, whereas unsigncryption has two EC-point multiplications and one EC-point addition. The running time is not affected by changing the AES encryption key, because the signcryption algorithm applies a SHA 256-bit hash function to the key before using it (see Equs. (2) and (3)). Thus, although keys with variable bit lengths are provided, encryption is always performed with the 256-bit key K_2. We intend to compares these results with state-of-the-art approaches in our future work.

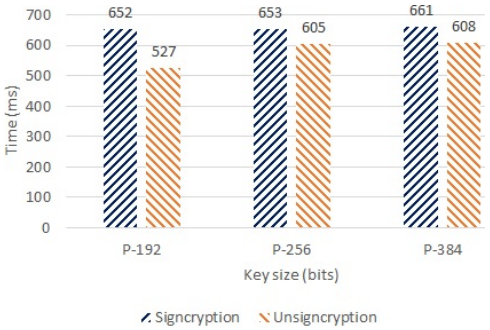

Figure 4. Running time of signcryption and unsigncryption with different EC keys for an 480×320 image with a size of 105 kB.

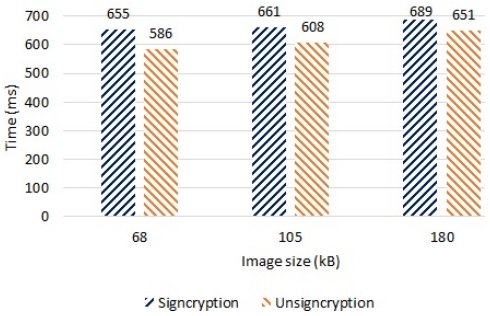

Figure 5. Running time of signcryption and unsigncryption with different image sizes using an EC P-384 key

A. Discussion

Due to the smaller key size and the single-step implementation of signature and encryption, EC-based signcryption has potential advantages over existing works such as the sequential implementation of watermarking [13] or RSA-based digital signature [7] with AES encryption. It was demonstrated [24] that a comparable security level can be obtained by EC using a smaller key length with respect to RSA (e.g., 160-bits key with EC cryptography is equivalent to 1024-bits key with RSA). Hence, the implementation of EC-based signcryption on the image or video frames requires less computational costs. The multiplication and addition operations of EC-points are the most time consuming parts of signcryption and unsigncryption processes. However, it is worth noticing that these parts need to be executed only once at the beginning of the signcryption process and after that, only encryption or decryption part influences the running time. A hardware accelerators for hash function, AES and EC on the smart camera can improve the computational efficiency.

VI. Conclusions

In this work, EC-based signcryption has been used to protect data captured by smart cameras for event-triggered monitoring in IoT applications. We first identified the potential threats for such applications and then analyzed selected security issues. The proposed signcryption, which is implemented on the sensing unit, provides countermeasures to the possible threats and enables the authenticity of encrypted images on the untrusted camera host part without compromising its confidentiality. We analyzed the running time of proposed signcryption techniques on Raspberry Pi-3. The results show that EC-based signcryption is resource efficient for the security of image or video frames directly on the sensing unit.

Our future plans include the exploitation of physical unclonable functions (PUFs) to generate secure and tamper-proof private keys for resource constrained sensing units. We plan to use ECDLP for generating the associated public

keys from that PUF-based private keys. Another direction is to extend this work for the security and safety of public premises. In our current work we initiated the data transfer when simple pre-defined events have been detected. Detection of more complex or "unusual" events requires substantial computation which might be challenging for resource-constrained sensing units. Another challenge for such scenarios is to maintain the privacy of the observed people.

ACKNOWLEDGMENT

This work has been supported in part by the Erasmus Mundus Joint Doctorate in Interactive and Cognitive Environments, which is funded by the Education, Audiovisual & Culture Executive Agency.

REFERENCES

[1] M. Reisslein, B. Rinner, and A. Roy-Chowdhury, "Smart camera networks [guest editors' introduction]," *Computer*, vol. 47, no. 5, pp. 23–25, May 2014.

[2] T. Winkler and B. Rinner, "Security and privacy protection in visual sensor networks: A survey," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 2:1–2:42, May 2014.

[3] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," in *Proc. IEEE Symposium on Security and Privacy (SP)*, May 2016, pp. 636–654.

[4] X. Zhou, "Improved signcryption scheme with public verifiability," in *Proc. KESE Pacific-Asia Conference on Knowledge Engineering and Software Engineering*, Dec 2009, pp. 178–181.

[5] T. Winkler, A. Erdelyi, and B. Rinner, "Trusteye.m4: Protecting the sensor not the camera," in *Proc. IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Aug 2014, pp. 159–164.

[6] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in embedded systems: Design challenges," *ACM Trans. Embed. Comp*, vol. 3, no. 3, pp. 461–491, Aug. 2004.

[7] T. Winkler and B. Rinner, "Secure embedded visual sensing in end-user applications with TrustEYE.M4," in *Proc. IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, Apr 2015, pp. 1–6.

[8] F. Li, Y. Han, and C. Jin, "Practical signcryption for secure communication of wireless sensor networks," *Wireless Personal Communications*, vol. 89, no. 4, pp. 1391–1412, 2016.

[9] V. M. Potdar, S. Han, and E. Chang, "A survey of digital image watermarking techniques," in *Proc. IEEE International Conference on Industrial Informatics*, Aug 2005, pp. 709–716.

[10] P. W. Wong, "A public key watermark for image verification and authentication," in *Proc. International Conference on Image Processing*, vol. 1, Oct 1998, pp. 455–459.

[11] M. Schneider and S.-F. Chang, "A robust content based digital signature for image authentication," in *Proc. International Conference on Image Processing*, vol. 3, Sep 1996, pp. 227–230.

[12] P. K. Atrey, W.-Q. Yan, and M. S. Kankanhalli, "A scalable signature scheme for video authentication," *Multimedia Tools and Applications*, vol. 34, no. 1, pp. 107–135, 2007.

[13] S. P. Mohanty, "A secure digital camera architecture for integrated real-time digital rights management," *Journal of Systems Architecture*, vol. 55, no. 10–12, pp. 468 – 480, 2009.

[14] G. R. Nelson, G. A. Jullien, and O. Yadid-Pecht, "Cmos image sensor with watermarking capabilities," in *Proc. IEEE International Symposium on Circuits and Systems*, May 2005, pp. 5326–5329 Vol. 5.

[15] P. Stifter, K. Eberhardt, A. Erni, and K. Hofmann, "Image sensor for security applications with on-chip data authentication," in *Proc. of the Society of Photo-Optical Instrumentation Engineers*, vol. 6241, pp. 8, Apr 2006.

[16] D. Serpanos and A. Papalambrou, "Security and privacy in distributed smart cameras," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1678–1687, Oct 2008.

[17] T. Winkler, "Demo: TrustEYE.M4, A novel platform for secure visual sensor network applications," *in Proc. International Conference on Distributed Smart Cameras*, pp. 1–3, Jan 2014.

[18] Y. Cao, L. Zhang, S. S. Zalivaka, C. H. Chang, and S. Chen, "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, Nov 2015.

[19] A. Jurisic and A. Menezes, "Elliptic curves and cryptography," *Dr. Dobbs Journal*, pp. 26–36, 1997.

[20] E. Mohamed and H. Elkamchouchi, "Elliptic curve signcryption with encrypted message authentication and forward secrecy," *International Journal of Computer Science and Network Security*, vol. 9, no. 1, pp. 395–398, 2009.

[21] I. Haider, M. Höberl, and B. Rinner, "Trusted sensors for participatory sensing and iot applications based on physically unclonable functions," in *Proc. ACM International Workshop on IoT Privacy, Trust, and Security*. ACM, 2016, pp. 14–21.

[22] J. Pacheco and S. Hariri, "Iot security framework for smart cyber infrastructures," in *Proc. International Workshops on Foundations and Applications of Self* Systems (FAS*W)*, Sep 2016, pp. 242–247.

[23] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[24] M. Yasuda, T. Shimoyama, J. Kogure, and T. Izu, "Computational hardness of ifp and ecdlp," *Applicable Algebra in Engineering, Communication and Computing*, vol. 27, no. 6, pp. 493–521, 2016.