Subhan Ullah, M.Sc.

Secure Camera Nodes for IoT Applications

DISSERTATION

to gain the Joint Doctoral Degree
Doctor of Philosophy (PhD)
Alpen-Adria-Universität Klagenfurt
Fakultät für Technische Wissenschaften

in accordance with
The Erasmus Mundus Joint Doctorate in Interactive and Cognitive Environments
Alpen-Adria-Universität Klagenfurt — Universitá degli Studi di Genova

First Supervisor
Prof. Bernhard Rinner
Institute of Networked and Embedded Systems
Alpen-Adria-Universität Klagenfurt, Klagenfurt, Austria

Second Supervisor
Prof. Lucio Marcenaro
Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture
Universitá degli Studi di Genova, Genova, Italy

March, 2019

Acknowledgments

This PhD Thesis has been developed in the framework of, and according to, the rules of the Erasmus Mundus Joint Doctorate on Interactive and Cognitive Environments EMJD ICE [FPA n^o 2010-0012] with the cooperation of the following Universities:

Alpen-Adria-Universität Klagenfurt - AAU

Queen Mary University of London - QMUL

Technische Universiteit Eindhoven - TU/e

Universitá degli Studi di Genova - UNIGE

Universitat Politécnica Catalunya - UPC

According to ICE regulations, the Italian PhD title has also been awarded by the Universit degli Studi di Genova.
This work was also supported by the research initiative Mobile Vision Austria with funding from the Austrian Federal Ministry of Science, Research and Economy and the Austrian Institute of Technology.

First Reviewer
Prof. Bernhard Rinner
Institute of Networked and Embedded Systems
Alpen-Adria-Universität Klagenfurt, Klagenfurt, Austria

Second Reviewer
Prof. Lucio Marcenaro
Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture
Universitá degli Studi di Genova, Genova, Italy

Affidavit

I hereby declare in lieu of an oath that

- the submitted academic paper is entirely my own work and that no auxiliary materials have been used other than those indicated,
- I have fully disclosed all assistance received from third parties during the process of writing the thesis, including any significant advice from supervisors,
- any contents taken from the works of third parties or my own works that have been included either literally or in spirit have been appropriately marked and the respective source of the information has been clearly identified with precise bibliographical references (e.g. in footnotes),
- to date, I have not submitted this paper to an examining authority either in Austria or abroad and that
- when passing on copies of the academic thesis (e.g. in bound, printed or digital form), I will ensure that each copy is fully consistent with the submitted digital version.

I understand that the digital version of the academic thesis submitted will be used for the purpose of conducting a plagiarism assessment.
I am aware that a declaration contrary to the facts will have legal consequences.

Subhan Ullah e.h. Klagenfurt am Wörthersee, March 27, 2019

Dedicated to my beloved parents and my wife.

Acknowledgments

I would like to praise and thank Almighty Allah for blessing me strength, determination, ability and the opportunity to complete this Ph.D. thesis. I would like to express my sincere gratitude to my supervisors Prof. Bernhard Rinner and Prof. Lucio Marcenaro for their unprecedented support, guidance, encouragement, constructive feedback with critiques and suggestions for the development of this research work. I would like to thank all my colleagues at the Pervasive Computing group for their ideas and feedback during Jour Fixe or other academic meetings and discussions.

I would like to thank Ms. Heidelies Aschbacher for administrative support and assistance throughout my Ph.D. work. I would like to thank EMJD ICE didactic managers, particularly Ms. Laura Zoppi for the assistance and support at the University of Genoa, Italy.

I would like to thank my parents, brothers, sisters and my wife and children for their enormous love, support, and prayers. Finally, I would like to thank all the people who made this research work possible for me.

Thank you all!
Subhan Ullah

Secure Camera Nodes for IoT Applications

Abstract

Smart cameras are expected to become key sensor devices for various Internet of Things (IoT) applications. Since cameras often capture highly sensitive information, data protection and security is a major concern. The basic challenges to the implementation of data protection and security approaches in smart cameras are resource limitation, processing of high volume data, open infrastructure and realtime performance. Resource efficient approaches are required to overcome these basic challenges and provide enough data protection and security. This thesis investigates elliptic curve (EC)-based signcryption approaches towards data protection and security for smart cameras with reduced computation and communication overheads. Signcryption achieves resource-efficiency by performing data signing and encryption in a single step. By running the EC-based signcryption on the trusted sensing unit, this work can relax some security assumptions for the camera host unit which typically runs a complex software stack. Smart cameras are resource-limited and a single camera cannot provide sufficient and reliable monitoring for complex environments in large areas, and often requires smart camera networks. The part of this thesis presents aggregate-signcryption and extends the deployment of EC-based signcryption approach to cluster-based smart camera networks. This aggregate-signcryption reduces the communication overhead and requires fewer steps for the unsigncryption as compared to the individual-signcryption. Further, this work introduces system architecture motivated by a typical case study for camera-based IoT applications, evaluates security properties and presents performance results of an ARM-based implementation. The part of this thesis also generalizes the aggregate-signcryption for a multi-sender/multi-receiver scenario. These approaches provide basic security and data protection with reduced computation and communication overheads and this thesis investigates resource consumption for all of these three approaches in their respective scenarios.

Contents

List of Figures

List of Tables

Glossary

G	The base point of the elliptic curve (EC).
F_q	EC finite field.
q	A prime number specifying the finite field.
n	Big integer, which represents the order of F_q.
v	A prime number where, $v \in \{2, 3, ...n - 1\}$.
K_{enc}	AES encryption key.
K_{dec}	AES decryption key.
Pu	Represents public key.
Pr	Represents private key.
i	Represents the identity of cluster head.
CH_i	Cluster head i.
j	Represents the identity of smart camera.
C_j	Smart camera j.
h	Represents the identity of monitoring device.
M_h	Monitoring device h.
k	Bit length of security parameters.
$\mathbb{Z}_q^+$	Set of additive integers over mod q.
$\mathbb{Z}_q^*$	Set of multiplicative integers over mod q.
g_1	Group field generated by EC with base point G.
x	Master secret key chosen by KGC.
k_0	Bit length of security parameters.
H	One-way and collision resistant hash function.
c	Represents ciphertext in individual-signcryption.
R	Part of signature in individual-signcryption.
s	Part of signature in individual-signcryption.
k_1	Represents the hash value of the result of $v{\cdot}G$.
r	The hash value of the result of c, k_1.
ω	Represents the internal state information of a device.
Pr_j	Private key of smart camera j.
Pu_j	Public key of smart camera j.
Pr_h	Private key of monitoring device h.
S	The sum of all signcryption parts of smart camera j.

σ	Random number chosen by smart camera j.
d	Represents the part of partial private key.
X	Represents the part of partial private key.
Pu_{kgc}	Represents the public key of KGC.
P_j	Public value of smart camera generated by using base point G.
$P_{j_{kgc}}$	Public value of smart camera generated by using Pu_{kgc}.
r_j	Secret value chosen by KGC for partial key generation of smart camera j.
$K_{enc(l)}$	Encryption keys generated on smart cameras for list l monitoring devices.
$K_{dec(l)}$	Decryption keys of list l monitoring devices.
s_j	Secret key chosen by smart camera e.g., $s_j \in \mathbb{Z}_q^*$.
θ	The combined data of individual-ciphertexts.
ϑ	Represents aggregate-signcryptext data.
$\varrho(l)$	Individual-ciphertext for h monitoring devices in l.
f	Number of frames for the region of interest (RoI).
l	List of the monitoring devices.
c'	Represents modified ciphertext.
U_h	Represents a part of signcryptext for the association of multi-receiver.
V	Represents a part of multi-receiver signcryptext.

Chapter 1

Introduction

Smart cameras are real-time video acquisition and processing systems that combine onboard sensing, processing and communication capabilities [173]. These devices have undergone tremendous advances over the last decade and play an important role in several IoT applications [109, 60, 126]. Smart cameras often perform sensing, processing and communication for visual surveillance and monitoring tasks (e.g., detection, capturing, identification and tracking etc.) as a stand-alone single smart camera or network of multiple smart cameras. Smart cameras are resource limited and a single smart camera cannot perform sufficient and reliable video surveillance for large area and complex IoT applications. The network of smart cameras is often necessary to cover all the locations of a large area in complex environments. Smart cameras perform the monitoring tasks in different communication scenarios ranges from single-sender/single-receiver to multi-sender/multi-receiver scenarios in IoT applications. Smart cameras either perform the surveillance tasks as a stand-alone single node or as a network play an important role in their respective application scenarios. However, security and privacy protection have become a major concern due to their widespread deployment, the sensitive nature of the captured data and the open infrastructure [170, 53]. 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. The implementation of data protection and security for single-sender/single receiver scenario can be easily manageable due to the individual setup, however the complexity of techniques and security requirements are increasing for multi-sender and multi-receiver scenarios. In a multi-sender/single-receiver scenario a group of smart cameras perform video surveillance tasks at the same time for a specific location and transfer the captured information to a single receiver (monitoring device). The complexity of security techniques in multi-sender/single-receiver scenario is the verification of large volume of data (video or images) received from a group of cameras at the same time on a single receiver (monitoring device). This large data verification has a bottleneck on a single receiver. The complexity is further increases if a group of smart cameras requires to share the same information with more than one monitoring devices (e.g., in the multi-sender/multi-receiver scenario). The complexity of the security techniques and data protection approaches for multi-sender/multi-receiver scenario are the sharing of same (e.g., one time protected) data to multiple receivers with exclusive access for all of them. A decryption fairness property of the protected data is required for multiple receivers to exclusively access the same data. The basic challenges in the deployment and implementation of data protection and security approaches for smart cameras are the resource limitations, processing of high volume data (images/videos), deployment in open infrastructure and realtime performance. Resource efficient data protection and security approaches are required to overcome these challenges and provide the protection and security of data with reduced computation and communication overheads.

1.1 Motivation and Objectives

This doctoral study introduces a security approach for smart cameras by integrating signcryption [95] with EC for improving resource efficiency. The thesis has extended the preliminary work [167] on securing the camera node by separating the platform into a trusted sensing unit with exclusive access to the image data and an untrusted camera host unit which running user specific applications, operating system, middleware and networking tasks. Such separation helps to mitigate the increasing attack threats for complex embedded software systems [125]. Integrity, authenticity and confidentiality of data 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. [171]). Signcryption is a resource-efficient technique which implements signature and encryption in a single step with reduced computational and communication costs as compared to traditional sequential approaches [84, 192]. This work applies EC-based signcryption directly on the sensing unit in order to push data protection possibly closed to the visual sensor. The particular challenges for this thesis approach are the resource limitations, the processing of high volume of image or video data, the open infrastructure (e.g., Internet) and real-time performance in IoT application. The EC-based signcryption technique has been proposed as an efficient solution for securing the smart camera by separating the platform into a trusted sensing unit and an untrusted camera host unit [167]. In that case, security was ensured by individually protecting images transmitted from the smart camera. In an IoT environment more than one smart camera is often required to monitor a wide area [77]. The transmission of signcrypted images or video frames from co-located distinct smart cameras at the same time might saturate the communication channel and overload the end-user device during the verification and unsigncryption process. A part of this thesis proposes a cluster-based multi-camera architecture (Figure 3.4) which is able to efficiently process and secure the captured data. This thesis divides the network of smart cameras into distinct clusters and extends the EC-based signcryption [154] to an aggregate-signcryption approach. The aggregate-signcryption is entirely performed on a cluster head by merging the signcryptexts and corresponding public keys of the smart cameras in the same cluster. Aggregate-signcryption combines signcryptexts to reduce the signature data without losing any security properties of the individual signcryptexts. The aggregate-signcryption saves communication costs during its transmission and the computation resources during its verification on the monitoring device (e.g., smartphone). The clustering approach provides scalability and management to the network of smart cameras, where aggregate-signcryption provides an efficient approach in term of computation and communication for data protection.

As, the effectiveness and efficiency of the protection techniques is a particular challenge due to the resource limitations of the smart-camera devices, this thesis addresses the resource limitation and introduces a lightweight security approach for smart camera IoT applications. In the first place, the thesis deployed individual- and than the aggregate-signcryption for protecting sensitive data in smart camera networks and secured the transfer from a single smart camera or clusters of smart cameras to a single monitoring device [154, 155]. Then, this thesis generalizes the aggregate-signcryption approach to efficiently protect the data from multiple cameras to multiple monitoring devices. These approaches provide integrity, authenticity and confidentiality of data with decryption fairness for multi-receiver throughout the entire lifetime of the data. It further provides public verifiability (i.e., any trusted party can verify the authenticity) and forward secrecy of data (e.g., the confidentiality of incoming data will not compromise if the current or past session (encryption) keys has been compromised by someone) .

1.2 Scientific Contributions

The main scientific contributions of this thesis can be summarized as follow:

Deployment and evaluation of EC-based signcryption. The first contribution lies in the deployment and evaluation of the EC-based signcryption directly on the sensing unit. This thesis proposes the overall system architecture which is motivated by a smart home surveillance case study and briefly analyzes security properties of the proposed 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 monitoring devices and some backup server. The thesis further presents runtime measurements on an ARM-based implementation [154].

Deployment and evaluation of aggregate-signcryption. The second contribution lies in the deployment and evaluation of EC-based aggregate-signcryption in a cluster-based smart camera networks in IoT applications. This thesis work enables the cluster head to efficiently apply aggregation on the collected signcryptexts of distinct co-located smart cameras. The proposed architecture has been implemented and evaluated on a network based on Raspberry Pi nodes [155]. The security of the aggregate-signcryption is analyzed and the performance ratio in term of communication and computation are discussed.

Generalization of aggregate-signcryption. The third contribution includes the deployment of EC-based aggregate-signcryption for multi-receiver in an IoT scenario. This work adopts a certificateless approach by using a KGC and avoids the key escrow [176] problem of key sharing and authentication. This certificateless multi-receiver aggregate-signcryption approach has been implemented and evaluated in a smart-camera IoT scenario. The proposed approach saves 32.89% and 28.90% of the computational time as compared to the individual- and aggregate-signcryption in a multi-sender/multi-receiver scenario [153].

Investigation of resource-consumption. This thesis also investigates the resource consumption of all the above three approaches and presents its evaluation and comparisons.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

Chapter 2. Chapter 2 of this thesis discusses state-of-the-art approaches for smart camera IoT applications. The main focus of these approaches is the implementation of data protection and security techniques for resource-limited smart cameras. The chapter reviews the smart camera systems, its types, platforms and video surveillance applications. The chapter highlights the security requirements, considerations, techniques, implementation and data protection approaches of smart camera systems. The chapter also discusses the state-of-the-art secure smart camera use-cases, introduces the proposed EC-based signcryption approaches and compares them with the state-of-the-art approaches.

Chapter 3. This chapter presents the overall approach of data protection and security techniques proposed and implemented as part of this thesis. The thesis introduces monitoring use-case and describes the system architecture and its scenarios in this chapter. The chapter also highlights a threats model and identifies the possible attacks scenarios. The chapter provides a brief and overall description of security requirements and assumption for smart cameras security and summarizes the proposed techniques and approaches.

Chapter 4. Chapter 4 introduces the EC-based signcryption (individual-signcryption) for data protection and security onboard the smart camera. The chapter presents the signcryption and unsigncryption algorithms and security analysis. The chapter also describes the experimental setup and discuses the evaluated results.

Chapter 5. This chapter of the thesis presents the aggregate-signcryption approach, its deployment and operational phases for the multi-sender/single-receiver scenario of smart cameras in IoT applications. The chapter provides a security analysis of the aggregate-signcryption and evaluates the results of the performance ratio in term of computation and communication.

Chapter 6. Chapter 6 explains the generalization of aggregate-signcryption and discusses the multi-receiver aggregate-signcryption. The chapter describes the deployment and operational phases of the data protection and security of the multi-receiver aggregate-signcryption approach. The chapter analyzes the security of multi-receiver aggregate-signcryption for multi-sender/multi-receiver scenario and evaluates the experimental results. The chapter also compares the performance of individual- aggregate- and multi-receiver aggregate-signcryption for multi-sender/multi-receiver scenarios.

Chapter 7. Finally, Chapter 7 concludes the thesis and provides an outlook for future possible research work in the investigated field.

Chapter 2

State-of-the-Art

Overview

This chapter discusses state-of-the-art research work relevant for this thesis. The main objectives of this thesis is the protection and security of data (images/video) captured by smart camera systems either in the context of IoT or visual sensor networks (VSN) applications [37, 170]. The main focus of these applications is video surveillance of private [61, 171] or public premises [82]. First, this chapter reviews the smart camera systems and their relevant applications in the context of IoT or VSN. The smart camera in IoT or VSN applications has the challenges of resource limitations, realtime performance and processing of high volume data (e.g., images or videos). Second, the chapter discusses data protection and security in smart camera systems and identifies its basic requirements and consideration. The related approaches and techniques for data protection and security are discussed. The focus of these approaches and techniques is their efficiency and performance in the implementation to overcome the challenges of smart camera systems either fully or partially. Third, the chapter reviews potential use-cases of secure smart camera systems and highlights the security approaches implemented for the data protection and security. Forth, the chapter discusses the proposed approach of this thesis (signcryption based data (image or video) protection and security in smart camera systems). Signcryption approach is implemented to provide basic security and data protection while overcome the challenges of smart camera system in IoT or VSN applications. The chapter finally presents the comparison of the EC-based signcryption approaches with the state-of-the-art.

2.1 Smart Camera Systems

Smart cameras play important roles in a wide range of IoT applications [44], especially in video surveillance applications [10]. This discussion is mainly focused on those IoT and VSN applications where a smart camera often captures sensitive information in the form of image or video data for monitoring devices or end-user applications. Single stand-alone smart camera is used for simple applications [93] while multiple cameras (smart camera networks) are used for complex applications to fulfill the application specific requirements (e.g., video surveillance of large area).

2.1.1 Stand-Alone Smart Camera

A stand-alone smart camera can perform surveillance tasks independently and shares the captured data directly with the end-user monitoring devices. A stand-alone smart camera is usually a pinhole camera modeled with a perspective effect of projection [117]. A stand-alone camera with perspective projection is applicable for the detection and viewing of objects in one direction (e.g., face detection and identification of people on their entrance to home). However, the applications where surveillance are required in all directions or for the monitoring of large areas, the stand-alone or single camera is not sufficient because of their limited field of view.

The problem of this limited view can be solved with an omnidirectional smart camera (catadioptric camera or fisheye camera) [161, 105]. Omnidirectional smart cameras have hemispheric or complete spherical (360 degree) field of view with the horizontal plane [134, 135]. Meinel et al. [94] present a single omnidirectional camera as realtime surveillance system for ambient assisted living (AAL) which can monitor the entire room and track people (e.g., entry or leaving) in a room. Scharfenberger et al. [136] present a omnidirectional camera for the operations (closing or opening) of a door for a smart car. Their omnidirectional camera monitors and predicts the risk of collision with the door of an approaching car or cyclist from outside. This camera can track objects (e.g., human, cars, bicycles) and provides enhancement to the outside side mirrors of cars which are used by the drivers to look outside for the safety purposes before opening the door or crossing a lane. Omnidirectional smart camera systems are also used in robotic applications for the localization, movement and navigation of vision-guided robotes [120].

The limited view problem can also be solved by using multiple smart cameras for surviellance. Multiple smart cameras can view a targeted object from different view points and from different angles which further enhance the reliability and trust on the captured data [186, 142]. Multiple smart cameras are required to connect with each other to establish a network for the surveillance of a large area.

2.1.2 Smart Camera Networks

Smart camera networks obtain information form the targets in the form of video or images for the processing and analysis of tasks (e.g., tracking, detection or identification of a target objects). The efficient processing and analysis of tasks require configuration, localization, orientation and grouping of smart cameras before the deployment of camera network because changes to the network setup are usually expensive after its deployment in a wide area surveillance application [89]. The optimal solutions for efficient processing (data analysis) and decision making on the captured data in camera networks are very important in order to exploit the advantages of smart camera networks. Smart camera networks can be classified on the bases of data analysis and decision making into centralized, distributed or cluster-based networks.

In centralized networks all smart cameras transfer the captured data (video or images) to the base station (central node) for the processing and analysis to provide a collective decision and further actions. The advantage of centralized network is the data collection from multiple cameras and its analysis for the derivation of more robust and reliable collective decisions. However, smart camera captures high volume of data (images/videos) and its transfer to a base station for processing is more expensive than the local processing on smart camera and the transfer of the processed data [59]. For example, in centralized network multiple smart cameras transfer the image or video data of detection and tracking of an object to a base station and then the base station decides further that which smart camera can efficiently track that specific object. The transfer of high volume data (images/videos) in centralized smart camera networks requires large bandwidth and sufficient amount of energy. Such requirements (e.g., large bandwidth and energy consumption) affects the realtime performance and considering as the limitation of centralized smart camera networks. Another limitation of centralized smart camera networks is the dependency of a system on single entity (base station). The dependency on single entity can lead to a single point failure problem (e.g., if the base station fails, the services of entire camera network will stop). Updating centralized camera networks by adding new cameras is expensive which are also limiting the scalability.

Local processing (data analysis) and decision making for predefined events onboard the smart camera can transfer only the important information (e.g., region of interest (RoI) data). The transfer of only the RoI information decreases the utilization of bandwidth and reduces the communication overheads. An alternative model to a centralized network is a distributed network, in which the processing of data and decision take place on each smart camera independently. A distributed network reduces the data transfer requirements to a dedicated node (base station), while it does not exploit the advantages of multi-camera networks (e.g., a collective decision by analysis of data received from multiple smart cameras) [7].

A cluster-based network can exploit the advantages of multi-camera networks by dividing the network of smart cameras in manageable and scalable small groups of smart cameras called clusters. A dedicated node of a cluster known as a cluster head manages all smart cameras within a cluster. A cluster-based network can improve the robustness of the network and decrease the bandwidth utilization as compared to distributed and centralized networks [185]. In this thesis a system architecture of cluster-based smart cameras network is implemented, which provides scalability and exploits the advantages of multi-camera networks [155, 153]. The efficiency and capability of smart camera systems to overcome the challenges of IoT or VSN applications (e.g., realtime performance, resource limitations, and processing of high volume data) also depend on the camera platforms and their architectures.

2.1.3 Smart Camera Platforms

A smart camera is a key sensor for IoT and VSN applications. The efficient sensing, processing and communication capabilities of smart camera platforms realize the performance of smart camera systems. The efficiency of sensing, processing, and communication capabilities of smart camera platforms depend on the proper selection of compatible hardware/software components, configuration and algorithms (e.g., image/video analysis algorithms) [132]. The careful configuration and design of the smart camera platforms can overcome the challenges of IoT and VSN applications and reduce the cost of smart camera systems. Security algorithms and techniques are required in addition to the normal tasks (image capture, tracking, detection or identification of objects) of smart cameras to provide security and protection of data. Najjar et al. [8] briefly reviewed some VSN platforms (Cyclops, MeshEye, Vision Mote and MicrelEye) related architectures and challenges. They further highlighted the need of lightweight algorithms for image processing and identified the trade-off between algorithms performance and resource consumption (memory, processing, and power). This thesis focuses on the data protection and security approaches where the proper selection of smart camera platforms, network architecture and system configuration improve the performance efficiency in term of communication and computation. This section reviews some relevant state-of-the-art smart camera platforms and their architectures as follows.

The Cyclops [122] platform is an early smart camera with very limited resources. It has 128 kB of Flash memory on-chip with 4 kB of SRAM data memory. It has also an external RAM of 60 kB to enlarge the data memory to 64 kB. Its micro-controller unit (MCU) has a 8 bit RISC core processor clocked at 7.3728 MHz and requires 3.3 V supply voltage. The Cyclops platform uses a MICA2 mote for communication because it dose not provide an onboard networking facility. The image sensor of the Cyclops platform captures 24 bit of RGB color image in 352 × 288 resolution.

MeshEye [65] is another energy-efficient smart camera platform with onboard sensing, processing and communication capabilities and is specially designed for intelligent surveillance applications. It has an unique vision system with low resolution to continuously determine the position, range, and size of moving objects in its field of view. MeshEye is equipped with an Atmel AT91SAM7 processor with 64 kB of SRAM, 256 kB Flash memory, 256 MB of MMC/SD memory card, and a 2.4 GHz ZigBee-based radio frequency (RF) transceiver. MeshEye uses two image sensors, one with low resolution for the detection of objects and another with high resolution for capturing the image in a good quality.

MicrelEye [74] is equipped with an FPGA board, a 8-bit AT40K MCU and 36 kB SRAM on the same chip. MicrelEye also consists of OV7620 CMOS camera to capture the images. Pixel-based background subtraction techniques are used for the detection of people. The reconfigurable FPGA is used for the background subtraction by assuming a predefined fixed background frame. They used the extracted features of detected object as input for the state vector machine (SVM) to further classify the behavior of a human being. The MCU is used for the features classification tasks. The MicrelEye consists of LmX9820A Bluetooth transceiver and a 1 MB storage for the image frames.

A Vision-Mote [187] is used for the water conservancy. It is a CMOS imager consisting of a 32-bit Atmel 9261 ARM9 processor and 128 MB of Flash memory, 64 MB SDRAM, and ZigBee-based communication module. It runs the Linux operating system (OS) and uses OpenCV (a machine vision library) to capture images and apply the compression and other processing functions. Multiple motes can be used in the form of a network called Vision Mesh to aggregate and compress the images for the base station in a multi-hop network.

Winkler and Rinner [168] presented TrustCAM, a secure embedded smart camera prototype. TrustCAM consists of a dual-core processor (ARM Cortex A8 CPU running at 480 MHz and a digital signal processor (TMS320C64x) clocked at 360 MHz), 256 MB RAM and 256 MB flash memory. TrustCAM uses a color SVGA CMOS image sensor (Logitech QuickCam Pro 9000), a

WiFi adapter (RA-Link RA-2571 802.11b/g) and XBee radio channel of low performance. TrustCAM further uses a readily available Atmel (AT97SC3203S) trusted platform module (TPM) chip connected via I2C bus with the mainboard of TrustCAM for the security and privacy protection of the captured data.

Winkler and Rinner [172] also presented a novel platform TrustEYE.M4 [1], which is equipped with a custom designed board with an ARM based Cortex M4 CPU (STM32F417) running at 168 MHz, 192 kB SRAM and 1 MB Flash memory on-chip. An additional 4 MB external SRAM is added to store multiple images and intermediate results of image processing algorithms because the on-chip SRAM is insufficient for such tasks. TrustEYE.M4 uses an image sensor (Omnivision OV5642) connected with TrustEYE.M4 board via an easily exchangeable dedicated port. The TrustEYE.M4 is equipped with a redpine signals RS9110-N-11-24-02 WiFi 802.11b/g/n radio. A secure communication can be established with end-user devices by using NFC interface. A security IC (ST33TPM12SPI) is also integrated for the data security and privacy protection techniques for VSN and IoT applications [171].

Birem et al. [28] introduced DreamCam, an FPGA-based smart-camera prototype which was proposed for real-time detection and extraction of visual information. The system architecture of DreamCam consists of five interconnected boards (FPGA Cylcone-III, image sensor, power, memory and communication), where the main one is the FPGA board, which is used as system on programmable chip (SOPC). DreamCam uses two types of image sensor boards (MT9M031 and EV76C560) with a similar electronic architecture. The MT9M031 board has 1.2 Mega pixel CMOS image sensor with the operational capability of 45 fps (1280 × 960 pixel resolution) or with reduced field of view (FOV) of 60 fps (at 720 HD resolution). The EV76C560 board has a CMOS active pixel sensor with 1.3 Mega pixel (1280 × 1024) dedicated for industrial uses. The Power board provides initial input voltage of 6.5 V and it varies from 1.2 V to 5 V for different types boards. The memory board of DreamCam also consists of six asynchronous SRAMs (each of 16 Mb) with controllable high-speed access time on 3.3 V. The communication board manages all the communications with a high speed USB 2.0 or Giga-Ethernet.

Rusci et al. [131] presented a fully programmable smart camera platform consists of low power FPGA (IGLOO nano AGLN250V2) board, control unit (CU) and a data path (DP) module. Their platform enables an efficient data processing by coupling the image sensor with multi-processing system of parallel ultra low power (PULP) digital processing unit [130]. Their smart camera platform also consists of 64 kB L2 memory, 48 kB tightly coupled data memory (TCDM) and other IO interfaces (SPI, UART and I2C). The image sensor of their smart camera platform captures 10 fps at (128 × 64) resolution. They used their platform for object detection, tracking and event triggering.

Haider and Rinner [61] presented a prototype with OV5642 5MP CMOS image sensor. Their prototype uses Zynq7010 SoC clocked at 666 MHz equipped with 1 GB DDR3 SDRAM and gigabit Ethernet communication interface. They also used physical unclonable functions (PUF) to extract the fingerprints of CMOS image sensor for the image security on the sensing unit. They used the prototype for private space monitoring in IoT applications.

Abas et al. [5] designed the SlugCam prototype with readily available off-the-shelf energy efficient components in order to reduce the cost and allows rapid development. SlugCam is a solar-powered smart camera with an extra rechargeable battery and used for outdoor video monitoring in IoT applications. The extra battery is used in the case when sunlight is not available. SlugCam is equipped with Raspberry Pi model B+ board [2] and an energy efficient microcontroller (MSP430). Raspberry Pi camera module can capture color images of 1920 × 1080 resolution with 30 fps maximum rate. SlugCam used WiFly (RN-174) module for communication. SlugCam also consists of a web-based server for storing video data and an interface for the end-users interaction to retrieve the data and manage the node remotely.

Lim et al. [85] designed and implemented an energy efficient IoT smart camera called CamThings. The CamThings smart camera uses a periodic on-off scheduling technique with the transmission of only selected and necessary images to a cloud server. CamThings consists of two microcontrollers, a main-MCU dedicated for wireless communication and a sub-MCU integrated to captures the images. These two microcontrollers turn off the power periodically after performing their dedicated tasks to save power consumption. For example, first the sub-MCU captures the images of the region of interest, forwards it to the main-MCU and turns off the power. Second, the main-MCU transmits the images through a gateway and then goes into sleep mode.

This thesis proposes an overall system architecture which is motivated by a smart home surveillance case study (event-triggered monitoring) where smart cameras perform onboard event detection and initiate the transfer of protected data to monitoring devices and some backup servers. The thesis further presents runtime measurements on a Raspberry Pi-3 platform 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 thesis also used standard laptops (Intel core i5 with 2.6 GHz and 8 GB RAM) running Windows 10 serve as platforms for further experiments. The proposed security approach is implemented on standard laptops for the ease of implementation and fair comparison of the different approaches.

Table 2.1 shows the comparisons of platforms which focus on the basic challenges to achieve efficiency and security for smart camera platforms or prototypes. The ✓ and ✗ represent yes for focusing and no for not focusing the relevant property, respectively. All these platforms are resource limited and the focus of these approaches are on the optimization techniques to overcome the basic challenges and achieve the efficiency and security.

Table 2.1: Comparison of smart camera platforms for challenges, efficiency and security. Legend: RL: resource limitation, RP: realtime performance, OI: open infrastructure HD: high volume data, CM: communication, CP: computation.

Platforms/Prototypes	Focused Challenges				Efficiency		Security
	RL	RP	OI	HD	CM	CP	
Cyclops [122]	✓	✗	✓	✗	✗	✗	✗
MeshEye [65]	✓	✗	✗	✓	✓	✓	✗
MicrelEye [74]	✓	✗	✓	✓	✓	✓	✗
Vision-Mote [187]	✓	✓	✓	✓	✓	✓	✗
TrustCAM [168]	✓	✓	✓	✓	✓	✗	✓
TrustEYE [172]	✓	✓	✓	✓	✓	✓	✓
DreamCam [28]	✓	✓	✓	✗	✓	✗	✗
Rusci et al. [131]	✓	✓	✓	✓	✗	✓	✗
Haider et al. [61]	✓	✓	✓	✗	✗	✓	✓
SlugCam [5]	✓	✗	✗	✗	✓	✗	✗
CamThings [85]	✓	✓	✗	✓	✓	✓	✗
This thesis [154, 155, 153]	✓	✓	✓	✓	✓	✓	✓

2.1.4 Video Surveillance Systems

Video surveillance is an important IoT and VSN application and is often used for indoor and outdoor monitoring of public or private premises. Single smart camera (as stand-alone smart camera) or network of smart cameras are used to perform surveillance tasks according to the requirements of a specific application. A generic architecture of video surveillance consists of smart cameras, backup servers and end-user monitoring devices [118, 61]. The maximum coverage of all the target locations of a surveillance area by smart camera networks is an important issue [20] and various techniques are used to model it appropriatly [91, 33]. The proper placement of smart cameras within network is also important to maximize the coverage and reduce the cost for achieving an efficient surveillance of a target location [19].

Multiple cameras with overlapping field of views can ensure higher robustness of camera networks and can monitor a targeted location from multiple points. A zoom adjustment of smart cameras can also change the field of view, however it has a trade-off for the coverage of large areas with the capturing quality of the image or video of target object, as the image quality is based on the distance of object from a camera and on the current zoom of a camera. Arezoo et al. [157] presented a resolution-based detection approach to determine the zoom of individual smart camera for the best possible coverage with highest quality of the image data of a targeted object. Wang et al. [162] presented a network of drone cameras for the surveillance of a sports field. They used a central node on the ground as an edge-node [133] to control the movement of drones for the next location and to assign an available associated base station. The central node ensures the maximum coverage of the scene and the highest bit-rate (throughput) of video streaming to associated servers.

It is important for a smart camera network to transfer only the necessary data, known as RoI data in resource-constrainted IoT-based surveillance applications [109, 40]. The selection and extraction of RoI from captured image or video data depends on the requirements of an end-user application (e.g., in smart home monitoring the slection of face images for people identification or the image of vehicle number-plate in traffic monitoring). As smart cameras capture large volumes of data in the form of images or video, the transfer of selected RoI information can reduces the communication overhead. The local processing and analysis of RoI data on the smart camera node reduce the communication overhead and privacy risks [127]. Smart camera surveillance can be event-driven (e.g., capturing of data upon event detection) or user-driven (e.g., capturing of new data or transfer of store data upon end-user requests) as compared to the traditional CCTV video surveillance systems [6, 4]. The sharing of resources among nodes in IoT applications can reduce the computation costs and preserve local resources only for critical processing [100]. Chien et al. [37] divided a network of distributed smart cameras into multiple groups and used a special node called aggregator in each group of smart cameras for aggregation, processing and analysis of the video data received from that group of smart cameras. The aggregator then only sends the necessary data (processed and analyzed data) to a monitoring device.

Kokkonis et al. [78] presented an IoT-based realtime surveillance system by using a 3D smart camera (also called time-of-flight or depth camera). They included the features of high efficiency video coding (HEVC) to ensure an efficient delivery of data. They deployed and analyzed frame synchronization techniques for realtime video streams. They proposed a novel transmission protocol (named NAMRTP) for a reliable data transfer over time-varying networks. In video surveillance systems the smart cameras capture highly sensitive information, the security of that information is important. Hence, energy efficient security techniques are necessary to protect the sensitive image or video data.

2.2 Data Protection and Security in Smart Camera Systems

Smart camera systems in IoT applications either consist of stand-alone smart cameras or smart camera networks and often capture, process, transfer and store data (images, videos, descriptive messages or RoI) [102]. This data usually consists of sensitive and personal information and vulnerable to several security attacks, which can be external or internal, passive or active, physical or logical, interruption or eavesdropping attacks [104]. These attacks can easily exploit resource limitations and open infrastructures (e.g., Internet) and compromise the origniality and correctness of data in smart camera systems [180]. The key objectives of a secure smart camera system are security and protection of data from all these attack scenarios. In smart camera systems the data (images, video and descriptive messages) protection and security is important to fulfill the requirements of a specific application. Smart camera systems in IoT applications need more energy/resource efficient solutions and techniques to fulfill the data protection and security requirements. This section first discusses the basic requirements and considerations of data protection and security of smart camera systems in IoT applications. Second, discusses the state-of-the-art approaches and techniques to fulfill those data protection and security requirements.

2.2.1 Requirements and Considerations

An ideal smart camera system provides data protection and security according to the surviellance needs of a specific application. The surveillance needs of a specific application depends on the context of an application scenario. Smart camera systems play important roles in various contexts of IoT applications (e.g., smart home monitoring, traffic monitoring and medical facilities etc.) and require data protection and security accordingly [24]. The data protection and security requirements depend on the context of an application scenario, for example in some scenario only the data correctness (integrity) is important and not the confidentiality or privacy of data. However in some scenarios the protection of data from unauthorized access is also necessary and if an attacker compromised the security of data without the knowledge of a concerned authority, then the data will not be acceptable and reliable for further use or decision making. The basic security and protection requirements for smart camera systems are the authentication, integrity, freshness, confidentiality and availability of data [140, 170]. Hence, ideal smart camera systems fulfill these protection and security requirements and prove the correctness, reliability, authenticity and protection of data throughout its life time, while they overcome the challenges of resource limitations, open infrastructure, high volume of data and realtime performance. An attacker can get access to IoT devices due to an open infrastructure of IoT applications (e.g., Internet) or insecure communication channel. They can use the identity of devices and send false data to end-user applications.

Authentication. Authentication is an important security property which provides the legitimacy of the source or sender of data [38]. It is important in smart camera systems to prove that the data is captured and sent by a known smart camera and not tampered by any attacker or unauthorized users. It is also possible that a legitimate source denies the ownership of their own data in future. Thus, additional techniques are required to prevent a source or originator from denying its own data later, which is called non-repudiation. A mutual authentication of communicating devices requires to choose a session key for the protection of data and to avoid such attacks [16]. The authenticity proof is also possible if a smart camera embeds some secret information (e.g., keyed hashing or signing of identity information) with image or video data. That secret information can provide the proof of authenticity on receiving side or on end-user applications [87].

Data Integrity. In IoT applications attackers can also alter the original data collected by smart cameras. They can alter the data stored on devices or during its transmission in a smart camera system. Hence, the implementation of data integrity [14] techniques is required to protect data modification by unauthorized users and attackers. Data integrity techniques provide the proof and assurance of data originality on the receiving entity or end-user applications [86]. The proof of originality is achieved by avoiding unauthorized access to data and implementing security and protection technique in the smart camera systems. Data protection and security techniques implemented close to visual sensing unit of smart camera are preferable approaches which can protect the data exactly on the origin (e.g., on visual sensor) [50].

Data Freshness. Data freshness [31] is also an important property for data security in smart camera IoT applications. Data freshness is a realtime assurance that the image or video data is not outdated and replayed. In a replay attack the same valid information is transmitted by an attacker repeatedly. Data freshness is an important security property related to data integrity and needs extra techniques (e.g., timestamps) to ensure the actual time of data generation (e.g., at what time image or video captured by a specific smart camera) on receiver side [69].

Data Confidentiality. An attacker or unauthorized user can also observe the personal information stored on a smart camera or during transmission across the network. This type of attack is called eavesdropping, which is usually a passive attack. In such passive attacks the attacker compromises the data privacy only and not the integrity of data [22]. Data confidentiality is required for data protection and security in smart camera system, which can protect the data from unauthorized users or attackers. The data confidentiality techniques (e.g., encryption) are required close to the visual sensing unit to avoid such attacks onboard the smart camera. The encryption techniques are required to ensure the confidentiality of data across the network as well as for the lifetime of data.

Data Availability. Accessibility and availability [129] of data and services to authorized users or devices are also required for an ideal smart camera systems in IoT applications. The accessibility of data only to authorized and authentic users and the protection from unauthorized users, is called access authorization of data. Data availability of smart camera systems is a guarantee to provide data and services whenever required to authorized users and provide the resistance against denial of service (DoS) attacks. Availability is also the assurance of accessibility of data for authentic and authorized users at any point of time.

Most of these data protection and security properties of smart camera systems are partially interdependent. For example, compromising the authenticity of image or video data loses the trust and reliability on its integrity and confidentiality. Similarly, the accessibility of data to authorized users and the availability of data are interdependent properties (e.g., preventing data accessibility from authorized users are in another words is the unavailability of data). Hence, lightweight/efficient data protection and security techniques/approaches are required to achieve these requirements, while considering the challenges of IoT applications (e.g., resource limitations, realtime performance and processing of high volume of data).

2.2.2 Protection Techniques and Approaches

This section discusses the state-of-the-art protection/security techniques and approaches for data (image or video) captured by smart camera systems in IoT applications. The focus of the discussion covered in this thesis is cryptographic techniques [123, 146] for smart camera systems but it also discusses some necessary non-cryptographic techniques [121, 39] for smart camera systems. Cryptographic techniques consist of algorithms (crypto protocols) and keys based on mathematical models called cryptosystems or cipher systems

[110]. Cryptosystems have wide range of implementation techniques for data protection and security in IoT applications. Cryptosystems are primarily used for the implementation of encryption and digital signature to provide confidentiality, authenticity and integrity of data among communicating parties (senders and receivers). The communicating parties share the keys (encryption keys or signing keys) with each other for encryption or signing of data. There are two main types of cryptosystems based on key sharing procedure, a symmetric cryptosystem and asymmetric cryptosystem.

In a symmetric key cryptosystem the communicating parties share the same secret key (single key) for data protection (e.g., for data encryption and decryption). Each party securely keeps the shared key and performs the encryption and decryption functions for the protection of data using the same secret key. The symmetric cryptosystems are usually more secure and efficient in computation but have some basic limitations. The first limitation is the difficulty of key distribution problem for sharing the secret keys within a system using open infrastructure (e.g., Internet). The second limitation is dealing with large number of keys in networks which needs secure storage on resource constrained devices. Third, a symmetric cryptosystem does not provide non-repudiation of the data. A well known symmetric cryptosystem for encryption is advanced encryption standard (AES) [42, 43]. The AES encryption is more secure, lightweight and efficient encryption algorithm as compared to other encryption schemes [98] for the confidentiality of data. The encryption approaches [156, 75, 179] are used to provide image and video confidentiality. Another example of a symmetric key cryptosystem is keyed hash function called message authentication code (MAC) [27]. The MAC function provides authentication and integrity of data at the same time. The combined approach of MAC and encryption in the MAC-then-encryption [61] way provides authentication, integrity and confidentiality but cannot provide the non-repudiation property. Unkeyed hash functions are also used to provide only the integrity of data.

In asymmetric key cryptosystems or public key cryptosystems a communicating party choose its own secret key according to a pre-defined system setup [146]. Then each party generates a public key based on its own secret key to share with all relevant communication parties. Each communication party stores its own secret key (also known as private key) securely and shares the public key on a public channel (e.g., through Internet) with each other. The generation of public keys are based on mathematical hard problem (known as one-way function) which are easy to compute in one direction (e.g., to compute the public key) but very hard to reverse it (e.g., to get the secret key back). Asymmetric key cryptosystems overcome the limitations of symmetric key cryptosystems. The public key generation can be performed on each party independently or on a trusted third party called KGC. A KGC generates public parameters and master secret key. The KGC securely stores its master secret key and shares the public parameters with other devices in the system. The devices use the shared public parameters of KGC and initiate the key generation procedure. The common examples of asymmetric key cryptosystems are RSA [128], digital signature algorithm (DSA) [63] and ECDSA [71] digital signature schemes. The digital signatures provide non-repudiation, authentication, integrity at the same time. MAC is an alternative of digital signature, however MAC does not provide non-repudiation.

Asymmetric cryptosystems are computationally expensive as compared to symmetric cryptosystems for the same level of security because of the key pair (private and public keys) computation, however it overcomes the challenges of symmetric cryptosystems. Asymmetric key cryptosystems use different complex mathematical models (also called hard problems) [101] for the computation of the algorithms. The RSA cryptosystem uses hard problem based on the integer factorization problem (IFP) [99]. RSA uses large prime numbers to ensure the security of the scheme [178]. The DSA cryptosystem uses the exponentiation problem called discrete logarithm problem (DLP) [92] for the generation of DSA security scheme. ECDSA bases on the elliptic curve discrete logrithm problem (ECDLP) [51] using small numbers as compare to both RSA and DSA schemes. The ECDSA cryptosystem is computationally efficient and provides equal level of security as compare to RSA and DSA schemes [116, 181].

A digital signature provides integrity, authenticity as well as non-repudiation for image data. Schneider and Chang [137] 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. [23] also applied a digital signature scheme to detect spatial cropping and temporal jittering in a video stream. They used three hierarchical levels (frame, shot and video) of videos and converted the input video into shots which were then converted to frames. For each level a signature is generated; a master signature is then derived from the individual level signatures using a master key. Signing at different levels allows the authenticity and verification of each frames, shot and complete video according to the need of end-user application.

State-of-the-art techniques also use some non-cryptographic approaches for the protection and security of image or video data [73]. A steganography [52, 64] is used instead of encryption scheme which hides or conceals image or video data or the channels information [48]. An obfuscation technique is used for the scrambling of the data pattern to protect the privacy concerns data. Thavalengal et al. [151] presented different techniques for the obfuscation of iris pattern of human eyes in digital photographs and video for iris recognition technology. They scrambled iris pattern without destroying the quality of image. They prevented privacy concerned attacks for iris recognition (e.g., iris spoofing attacks) to avoid the matching of original image and photograph image for iris recognition software. Schwarting et al. [138] exploits obfuscation for the authentication and identification (e.g., as a fingerprint) of image sensor. Digital watermarking [119, 175] is a widely used approach for the integrity verification of image data [81], i.e., to detect any changes in the size or pixel values of an image.

In IoT applications usually, more than one security property is required to fulfill the data protection and security needs of end-user applications. Some state-of-the-art approaches combine more than one cryptographic techniques to achieve more security properties at the same time. For example, the sequential implementation of digital signature and encryption techniques as sign-then-encryption [11, 184] provides the security properties of both digital signature and encryption. Similarly, the sequential implementation of watermarking and encryption techniques [163] provides both security properties of digital watermarking and encryption. A signcryption is another approach which implements digital signature and encryption in a single step and provides their security properties simultaneously. These data protection and security techniques can be implemented onboard the smart camera or across the smart camera networks according to the need of a specific application.

Onboard Protection of Smart Cameras

A stand-alone smart camera [173] which performs sensing, processing and communication on a single platform required resource-efficient security techniques for data protection and security [170] onboard. A desirable approach for the protection of the sensed data on a smart camera is the implementation of security techniques close to the sensing unit [169].

Winkler and Rinner [172] presented a novel platform TrustEye.M4 [1] using a hardware based TPM security chip for onboard security and privacy protection. By using RSA digital signature and time-stamping techniques, they were able to prove non-repudiation and authentication for the captured data. They demonstrated the feasibility of data protection at the sensor level with it custom-designed TrustEye.M4 prototype which provides sensor-level privacy protection [167]. The authors further integrated a TPM into the camera node and implemented RSA and AES following a sign-then-encrypt approach to ensure security and privacy of data onboard. The limitations of their approach are the significant hardware overhead for resource constrained sensors using TPM and the computational complexity by implementing the security techniques in the sign-then-encrypt way using 2048-bit RSA keys. Another limitation of the

TPM-based approach is the invalidation of security proofs if the data modified by an authentic entity for the sake of communication or computation efficiency on the host unit of the smart camera or any later stages.

Nelson et al. [106] 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. [148] 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 a MAC. They also equipped the image sensor with a dedicated EEPROM to uniquely identify the imager. Serpanos and Papalambrou [140] suggested that the image sensor should be trusted to prevent the insertion of unauthorized nodes in a distributed smart camera system. Cao et al. [34] proposed a CMOS image sensor based on 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.

Mohanty [96] presented a scheme called cryptmark which is based on digital watermarking and AES 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.

Haider and Rinner [61] presented a FPGA-based prototype exploiting PUF for identity-based signature (IBS) at the sensor level. They further used a certificate-based approach for key generation and implemented the AES-128 and HMAC-SHA256 using the encrypt-then-sign approach to ensure the security of data. The limitations of their approach are the key escrow problem [30] and the overhead of identity-based certification. It is also computationally expensive due to the implementation of encryption and signature in two steps. The AES implementation with a 128-bit key might be vulnerable to attacks. Cao et al. [35] also proposed a PUF-based CMOS image sensor for sensor-level authentication of the data. The preliminary work of this thesis presents signcryption technique to implement EC-based signature and AES-based encryption in a single step onboard the smart camera [154] in a single-sender/single receiver scenario. The onboard signcryption provides end-to-end data protection and security while reducing the computation and communication overheads of the smart camera system as compare to sign-then-encryption approaches.

The onboard data protection and security for a stand-alone single smart camera and for a network of multiple smart cameras are equally important to provide more reliability and trust on captured data. However efficient techniques are required for smart camera networks to overcome the challenges of multiple smart cameras (e.g., processing of the large volume of data, the secure handover of tracking information, the realtime performance and the open infrastructure etc.).

Data Protection in Smart Camera Networks

A single smart camera is not enough for the monitoring of large area and cannot fulfill the requirements of surveillance tasks due to its unidirectional and limited field of view. However, a network of multiple smart cameras can monitor a large area. The proper optimization of network architectures for resource management [182] overcome the challenges (e.g., realtime performance, open infrastructure and processing of high volume data) of smart camera networks. Ma et al. [90] proposed various efficient compression and communication techniques for multimedia data in resource-constrained applications for smart camera networks. The AES based encryption provides efficient confidentiality and considered to be a lightweight security algorithm which is suitable for the hardware and software implementation in smart camera networks [79, 98]. The architectures of the smart camera network also affect the computation efficiency of security techniques. For example the bottleneck in centralized network on the base station affect the realtime performance and communication efficiency. The bottleneck of base-station in a centralized network is due to the verification of security properties of protected data received from multiple smart cameras. The single point failure problem is also difficult to avoid in a centralized smart camera network. A cluster-based architecture can overcome the limitations of centralized and distributed networks and can provide the scalability without the risk of single-point failures [185, 57]. The smart cameras are grouped in distinct clusters in a cluster-based network, each with a local cluster head. A cluster head provides additional in-cluster security [141] in smart camera networks. The in-cluster security is the prevention of cluster nodes (e.g., smart cameras) from external threats which can target the smart cameras within a cluster. This additional security can be achieved by implementing traffic filtering and anomaly detection approaches on a cluster head. The anomaly detection and traffic filtering can be realized by implementing rule-based security approaches to allow only specific requests (e.g., the requests of a known end-users). The specific requests are minimum communication of authentic (known devices) necessary to fulfill a desirable service. The filtration of authentic requests can be achieved by specifying the communication rules (e.g., allowing known IP addresses and ports with the communication direction). A cluster head acts as a firewall for the rest of the cluster nodes and reduces the risk of external attacks [46] because the cluster heads forward only the authentic requests and distinguish them from malicious attacks. The communication rules can be provided by certification authorities or can be defined by the user (e.g., the integration of user-defined rules in smart cameras before its deployment) [177].

An end-user application in cluster-based smart camera networks can also verify the protected data at once as received from a specific cluster of cameras. Natarajan et al. [103] summarized the related work of smart camera networks for object detection, tracking, security, privacy, coordination and control strategies for video surveillance applications. Winkler and Rinner [172] used the TrustEye.M4 [1] platform as a stand-alone smart camera in a distributed network for the security and data protection of IoT and VSN applications.

This thesis proposes cluster-based smart camera network to provide scalability and exploits the advantages of multi-camera networks. The thesis extendes onboard data protection (EC-based signcryption for onboard smart cameras) of single-sender/single-receiver [154] to a multi-sender/single-receiver setup [155], while reducing computation and communication overheads. The data protection and security in multi-sender/single-receiver is achieved by aggregate-signcryption techniques by proposing a cluster-based smart camera network. The thesis further generalized the aggregate-signcryption approach for multi-sender/multi-receiver setups with decryption fairness for multiple receivers while maintains the resource efficiency [153]. Stand-alone single smart camera and the network of multiple smart cameras have a wide range of secure use-cases in IoT applications and among them video surveillance is being widely researched use-case for the investigation of data protection and security techniques.

Security Approaches for Video Surveillance Systems

In a video surveillance system each smart camera captures large volume of data in the form of images or videos and required efficient and lightweight techniques for data protection and security [170]. Lightweight symmetric and asymmetric cryptographic security techniques [79, 115, 144] are usually used for data protection and security. The symmetric and asymmetric techniques (e.g., encryption, digital signature and hashing etc.) provide basic security properties of data in IoT applications (e.g., confidentiality, authenticity, non-repudiation and integrity of data).

Alsmirat et al. [18] presented a framework for a secure video surveillance system [17]. They used AES for data confidentiality and RSA for key distribution. The session key was further secured by HMAC-MD5 hashing and provided authentication and integrity of the video streams. This approach was implemented with the NS-3 simulator and the trade-off between communication delay and security was evaluated. The computation and communication overheads were reduced by encrypting the whole video frame instead of encrypting each data packet.

Winkler and Rinner [171] presented a solution for the secure use of public cloud storage for data archiving and delivering. They used the hardware-based TPM security chip for onboard security and privacy protection on the smart camera. They also used AES for encryption and the RSA digital signature for signing with time-stamping techniques, and proved confidentiality, integrity and authentication for the captured data.

Won et al. [174] presented a certificateless multi-receiver hybrid encryption scheme for drone-based monitoring services in a single-sender/multi-receiver communication scenario, where the drone sends the sensitive data privately to multiple smart objects. They used efficient certificateless signcryption tag key encapsulation mechanism (eCLSC-TKEM) for key sharing and a certificateless multi-recipient encryption scheme (CL-MRES) for the encryption of data for multiple receivers. They proposed a tag with key encapsulation mechanism (KEM) and data encapsulation mechanism (DEM) approach [41]. They used that Tag-KEM/DEM approach and adopted the Boneh et al. [29] key revocation scheme to revoke the users. The Tag-KEM/DEM approach generalizes the (KEM/DEM) approach. They used hybrid encryption scheme based on a KEM/DEM approach to secure the one-time symmetric key along with data. They proposed the aggregate-signcryption approach and a dual channel strategy for efficient communication. In their approach, the sender reuses a proposed random number to generate the symmetric key used for each receiver. Drones are equipped with a GPU to reduce execution time and optimize the batch verification of signature to speed up the verification procedure. The authors implemented the secure communication protocols on two kinds of medium (equipped with moderate-speed CPU) and high capacity (equipped with CPU as well as GPU) drones for the smart parking and traffic monitoring applications.

2.3 Use-Cases of Secure Smart Cameras

A smart camera is a key sensor for numerous IoT applications, and more frequently used in the medical field, assisted living, monitoring application, agriculture, transport and industrial environments. In such applications, the data received by end-user applications from a smart camera should be correct, authentic, original and reliable. In most cases, the confidentiality of the captured data is also necessary. In the following use cases, the thesis discusses the required security and data protection properties needed for each use-case.

Smart Home Monitoring. A smart home is the most important use-case of a smart camera and a smart camera can surveillance smart home remotely from anywhere [114, 58]. The smart camera captures very sensitive and personal images or videos in the context of smart home use-cases. These image or video data should be only available to an authorized person. The protection of data needs appropriate and efficient techniques to implement security and data protection onboard the camera and in the network. The confidentiality, authenticity, integrity and access authorization are the key security requirements for smart home monitoring applications.

Assisted Living. Assisted living or ambient assisted living (AAL) [54] is also an important use-case of a smart camera, in which elderly people can be monitored remotely. Assisted living aimed to help the elderly and disable peoples (with special need) to live independently with the assurance of help and assistance when needed [124]. In assisted living use-case a smart camera captures the image or video of the person if some unusual behavior or activity has been detected, for example, the falling of a person. In this use-case, privacy and confidentiality of personal data is a very important issue. Baby-care or baby-monitoring system [158] is another use-case of a smart camera in the assisted living context, in which the movement, activities, health condition of a baby can be monitor remotely [149]. Data integrity and authenticity is important for baby remote monitoring applications.

Health-Care Monitoring. Smart cameras are used in a wide range of health-care applications [36], especially for the diagnostic of different diseases which are not possible to observe directly with human eyes. In health-care, the one use-case is endoscopy in which a micro smart camera help a doctor to view the internal organs or vessels of a patient and determine any abnormal symptoms [66, 80]. The images taken from a smart camera during an endoscopy should be correct and reliable, so the doctor can take a right decision for further treatment. Hence, the authenticity, integrity, freshness of data is required for health-care monitoring applications.

Remote health-care monitoring or mobile health [147] monitoring is another use-case which provides direct access to diagnostic and other health-related services regardless of time and place. Remote health-care monitoring can empower patients and especially elderly peoples for self-care and for ease in access to medical services. In these applications smart cameras also play an important role by transmitting the images or video of organs of a patient or observing the health condition of a patient from remote [32]. In this use-case an image or video captured by a smart camera should be authentic, correct and reliable, so the doctor can take the right decision on the treatment. In this use-case, the record of a patient should be confidential and only be accessible to the assigned doctor.

Smart City. Smart city [97] is an important use-case for a smart camera in the IoT applications [88, 183]. Smart cameras play an important role in the surveillance and monitoring of public places, streets, roads, parks, shopping centers, infrastructures and houses in a city. They can help police, city administration and monitoring authorities to keep an eye on the city. They can detect unusual activities during crowd monitoring of protest or festival. In a smart city, the smart camera captures personal data and that should be protected by law [164] and only exclusive access should be provided to concerned authorities. Smart camera can also send alerts to the concerned authorities if some roadside car accident occurs. In a smart city the sharing of information exclusively among various authorities to perform their own duties is a big challenge. The privacy and decryption fairness security properties are required for smart city use-cases.

Smart Parking. Smart parking [112, 160] is a smart city related use-case for smart cameras in IoT applications. Smart cameras are used to identify the vacant place and provide realtime information and guidance for the slot availability and reservation to the drivers [56]. In this use-case, the data captured by smart cameras should be authentic and accurate to fulfill the application needs.

A secure and privacy-preserving (SecSPS) [15] IoT framework for smart parking was proposed to find vacant parking places in a city center and monitor the incoming and outgoing vehicles in the parking spots. The SecSPS framework provided the detection and availability of vacant parking locations with real-time guidance to the driver for its reservation. The authors proposed certificate-based RSA key establishment techniques and 128-bit AES encryption with a cipher block chaining (CBC) mode for the confidentiality and hashing for data integrity using a sign-then-encrypt approach. Their proposed framework is resilient to various security attacks and ensures data protection and device security for the users. They suggested EC-based certification as an alternative for resource-limited devices. The SecSPS framework using RSA with 2048 bit and AES with 128 bit in a sign-then-encrypt way. The proposed approach due to the large key size of RSA, security concerns of the weak AES key and the implementation in a sign-then-encryption way is not energy efficient and secure for IoT applications.

Huang et al. [67] presented a security scheme to preserve the privacy of parking reservation in an automated valet parking (AVP) application. Their scheme protects the privacy of location and the identity of drivers and prevents double reservation attacks, where the users can only make a single reservation at the same time. The users can choose the vacant parking place by themselves and the location obfuscation mechanism easily provides location privacy for this use-case. They used a cryptographic approach based on an elliptic curve with bilinear pairing and simulated their scheme in Java for comparing the communication and computation overheads with state-of-the-art use-cases.

Traffic Monitoring Traffic monitoring is another use-case for a smart camera. A smart camera is used in traffic monitoring for the identification and recognition of vehicles to help the law enforcement authorities during transportation [25]. The images or video captured by a smart camera in traffic monitoring is also needed to be authentic and reliable, so the law enforcement authorities can easily decide about the situation or traffic dynamics. The basic security requirements for traffic monitoring use-cases are privacy, authentication, integrity, and freshness of the data captured by smart cameras.

2.4 Signcryption Security Approaches

The sequential implementation of digital signature and encryption provides the security properties (e.g., authentication, integrity, non-repudiation, and confidentiality) of images or video data in a holistic way. However, the implementation of sign-then-encrypt way is a two steps process, and its disadvantage is the extra overhead involved in the separate processing of the signature and encryption procedures. A signcryption approach is efficient than sign-then-encryption which implements digital signature and encryption in a single step. Signcryption approaches provide the same level of security (authentication, integrity, non-repudiation and confidentiality) simultaneously with reduced computation and communication overheads. Zheng [191] for the first time presented the concept of digital signcryption. He implemented ElGamal [49] digital signature (a DLP-based signature) and encryption in a single step and achieved the security properties of both digital signature and encryption at the same time. Zheng and Imai [190] also presented signcrypton based on the elliptic curve and demonstrated its efficiency over sign-then-encryption approach. State-of-the-art presented various digital signcryption approaches [150, 193, 192, 70, 189] based on either RSA, DSA or ECDSA digital signatures with AES encryption. The signcryption approaches which based on ECDSA and AES are considering as lightweight and efficient in term of computation and communication [95].

Zhou et al. [193] presented a lightweight short version of EC-based signcryption scheme for resource-constrained devices in IoT applications.

The efficiency of signcryption in term of computation and communication also depends on the key generation and key management techniques. The symmetric cryptosystems are efficient in term of key generations, however secure key distribution is difficult in symmetric cryptosystems. The asymmetric cryptosystem or public key infrastructure (PKI)-based cryptosystem solves the key distribution problem, however it also needs a third party trusted entity called certification authority (CA) to provide certification of public keys and its authentication. The certification management is an extra overhead for the deployment of signcryption approaches. Hence, a certificateless [45, 26] key management scheme is required to overcome this extra overhead.

Boneh et al. [29] used an identity-based cryptographic approach and eliminated the need for certification, but the limitations of their scheme were the key escrow problem. The key escrow problem is caused by a third party such as private key generator (PKG), which generated the secret keys. The key escrow problem was then eliminated by using bilinear pairing-based certificateless cryptography [30]. However, the computation of pairing was not efficient for resource-limited devices, so a pairing free approach [139] has been proposed and implemented in a drone-based surveillance application. The pairing free approach also faces the user revocation problem if a physical attack occurs on the device. In such cases, the attackers can access current and future information of the devices.

Pang et al. [113] presented a novel multi-receiver signcryption scheme and preserved the anonymity of senders and receivers. They also provided public verification and decryption fairness of the data. They multiplied the public key of the sender by a random value to hide the identity of the sender and avoided the cross-comparison and joint conspiracy attacks. Their scheme protects the data from both external and internal attacks. Their scheme is based on the security assumptions of decision bilinear Diffie-Hellman (DHBP) and Gap-BDH approaches. They theoretically evaluated the efficiency of this scheme and proved its security by using a random oracle model.

The efficient signcryption generalization for multi-user setups is also important for the smart camera networks. Niu et al. [108] presented hybrid signcryption which secures multiple messages for multi-receiver in heterogeneous environments. They used different master keys and sent multiple messages from a sender using identity-based cryptography to multiple receivers in a certificateless system. They used hybrid encryption based on a KEM and a DEM to secure the one-time symmetric key along with data. Their approach provides insider security by generalizing the KEM to signcryption KEM and included authentication. They used a PKG and a KGC to calculate the pseudo-identities for the users in their system and generated the partial private keys. They used the pairing-based cryptographic library (Libpbc) of C programming for the implementation of the scheme. They proved confidentiality and unforgeability in a random oracle model.

Nguyen et al. [107] presented an EC-based certificateless signcryption scheme based on the implementation of a Korean certificate-based digital signature algorithm (KCDSA) signature and symmetric key encryption (SKE) for IoT environments. They used the random oracle model to prove the security of the scheme, which provides confidentiality, integrity, authenticity and public verifiability. They demonstrated the communication and computation efficiency on an emulated Wismote sensor platform using C programming. These security properties are also achieved with another signcryption approach without pairing, which can be used for identity-based cryptography (IBC) in IoT and wireless sensor network applications [152]. They used a Laptop-PC and Raspberry Pi platforms for the valuation of their scheme. These both schemes are not applicable to the multi-receiver scenario and cannot provide decryption fairness for more than one receiving device.

2.4.1 Comparisons with State-of-the-Art Approaches

The comparisons of the algorithms, implementation procedures, security properties and efficiency of this thesis work with state-of-the-art approaches are summarized in Table 2.2. The summary only presents those algorithms and their implementation procedures which are specifically applied for (image/video) data protection and security in smart camera systems. The security properties and efficiency (in term of computation and communication) with state-of-the-art approaches are also compared. The efficiency is compared with respect to the multi-sender/multi-receiver scenario. The last column of the table also shows the comparison for the possible scalability of the approaches (e.g., in term of data protection and security among senders and receivers). The first three rows of the table with references [61, 171, 15] show the comparison of algorithms which has two steps implementation procedure (e.g., sign-then-encryption). All the sign-the-encryption approaches of the table are required certification techniques for public key authentication and do not provide the public verifiability. They are not efficient for multi-sender/multi-receiver scenarios and the data protection and security is possible for single-sender/single-receiver (e.g., scalability for 1-1) scenario. The next three rows with references [107, 152, 154] show the comparisons of signcryption approaches. The signcryption approaches provide all the basic security properties with reduced computation and communication overheads as compared to sign-then-encryption approaches. However, these signcryption approaches are not efficient and scalable for multi-sender/multi-receiver scenarios. The last four rows with references [145, 174, 155, 153] show the comparisons of aggregate-signcryption approaches. These aggregate-signcryption approaches also provide basic security properties with public verifiability. In these aggregate-signcryption approaches, this thesis presents an efficient (in term of computation and communication) certificateless multi-sender/multi-receiver approach [153]. This multi-sender/multi-receiver approach is also the contribution of this thesis and provides decryption fairness and access authorization for more than one receivers. The symbols (✓) and (✗) in the summary table shows yes and no respectively for the achieving of required properties. The 1 represents that decryption fairness and access authorization is only possible for a single receiver while > 1 shows the possibility of decryption fairness and access authorization for more than one receiver. The last column of the table shows the computation and communication efficiency of these approaches in the multi-sender/multi-receiver scenario. The last column also shows the scalability, where M represents (many) and the 1 represents a single entity (e.g., sender or receiver). All the other columns of the table are self-explanatory.

The overall summary shows that all the related work mentioned in this table provides the basic properties of security (e.g., authentication, integrity and confidentiality). However, this thesis achieves them efficiently as compared to state-of-the-art approaches. This thesis work proposed the EC-based signcryption techniques and 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 [154]. The smaller key size of EC [72, 83] and the implementation of signature and encryption in a single step [95] supports real-time data security directly on the sensing unit. This work further introduces aggregate-signcryption [155] to merge signcrypted data within a cluster of smart cameras and to extend the protection to a multi-sender/single-receiver setup. Each cluster of the smart cameras consists of a cluster head as an aggregator for all the signcryptexts of that cameras to further reduce the computation and communication overheads [155]. Finally, this work generalized the aggregate-signcryption techniques to multi-sender/multi-receiver setups while maintaining resource efficiency [153]. The these adopted a multi-receiver encryption scheme [165, 166] with a sign-then-encrypt approach and customized it to aggregate-signcryption with decryption fairness for multiple receivers. The proposed approach avoids the key escrow problem and does not require certification for public key authentication.

Table 2.2: Comparison of the thesis work with the state-of-the-art approaches for the security and efficiency acquired multi-sender/multi-receiver scenario. Legend: CL: certificateless, A: authenticity, I: integrity, C: confidentiality, DF: decryption fairness, PV: public verifiability, AU: authorization, CP: computation, CM: communication, SC: scalability.

Ref.	Algorithm	Implementation procedures		Security properties						Efficiency		
		Approach	CL	C	I	A	DF	PV	AU	CP	CM	SC
[171]	RSA, AES	sign-then-encryption	✗	✓	✓	✓	1	✗	1	✗	✗	1-1
[61]	HMAC, AES	encrypt-then-sign	✗	✓	✓	✓	1	✗	1	✗	✗	1-1
[15]	RSA, AES	sign-then-encryption	✗	✓	✓	✓	1	✗	1	✗	✗	1-1
[107]	KCDSA, SKE	signcryption	✓	✓	✓	✓	1	✓	1	✗	✗	1-1
[152]	ECDSA, SKE	IBC, DH, signcryption	✗	✓	✓	✓	1	✗	1	✗	✗	1-1
[154]	ECDSA, AES	signcryption	✗	✓	✓	✓	1	✓	1	✗	✗	1-1
[174]	eCLSC, CLDA	agg.signc, TKEM/DEM	✓	✓	✓	✓	1	✓	1	✗	✗	M-1
[145]	AES, BLS	obf-agg.signcryption	✗	✓	✓	✓	1	✓	1	✗	✗	M-1
[155]	ECDSA, AES	agg-signcryption	✗	✓	✓	✓	1	✓	1	✗	✗	M-1
[153]	EC-ShDSA, AES	agg-signcryption	✓	✓	✓	✓	> 1	✓	> 1	✓	✓	M-M

Chapter 3

Secure Smart Camera IoT Applications

Overview

A smart camera captures personal and sensitive data, and the implementation of resource efficient security techniques is a major concern. This thesis explores resource efficient data (image/video) protection and security techniques for smart camera IoT applications. These resource efficient security techniques are based on elliptic curve (EC) signcryption (cp. Chapter 4), aggregate-signcryption (cp. Chapter 5) and multi-receiver aggregate-signcryption (cp. Chapter 6) approaches. The signcryption approach implements digital signature and encryption in a single step which provides authentication, integrity, non-repudiation and confidentiality simultaneously. The smaller key size of elliptic curve and the implementation of digital signature and encryption in single step provide resource efficiency for smart camera IoT applications. This thesis measures the resource efficiency of data protection and security techniques in terms of computation and communication overheads. The efficiency of these security techniques also depends on the proper deployment of system architecture, data processing approaches and security algorithms.

This chapter presents an overall approach of the implementation and evaluation of signcryption, aggregate-signcryption and multi-receiver aggregate-signcryption security techniques. Section 3.1 briefly explains the overall approach of this thesis with the help of a monitoring use-case and introduces the system architecture, data processing approaches and the implementation of security techniques/algorithms. Section 3.2 presents the detail of system architecture and its communication scenarios. Section 3.3 identifies threats and attack scenarios. Section 3.4 illustrates the requirements and assumptions. Finally, Section 3.5 introduces the security and data protection approaches/techniques for each scenario according to the proposed system architecture.

3.1 Monitoring Use-Case

The primary goals of this thesis work is the implementation and evaluation of proposed security and data protection approaches in smart camera IoT applications while reducing the computation and communication overheads. This section presents the monitoring use-case with the help of a smart camera video surveillance system, which motivates this work for the implementation of efficient security and data protection approaches. In particular, this thesis envisions event-triggered monitoring in a smart home IoT application as a use-case. This event-triggered monitoring use-case has been deployed in ambient assisted living (AAL) scenario to monitor the activities of elderly people for their independent living and automated assistance. In this use-case, a smart camera detects some specific predefined events (e.g., fall detection of a person) in the AAL scenario and captures its (image/video) data. The smart camera processes that data onboard and then transmits real-time event alerts to an end-user monitoring device (e.g., caretaker smartphone). Due to the smart camera resource limitation, it processes only the region of interest (RoI) (image/video) data and upload it to a backup server for the long term permanent storage. The backup server sends a push notification to the monitoring device upon data receiving from smart cameras. The monitoring device sends a request to the backup server for the accessing of that relevant stored data. The backup server provides authorized access and forwards the requested data to the monitoring device. The architecture and data flow of the AAL scenario is shown in Figure 3.1.

The captured data of smart cameras in the AAL scenario consists of personal and privacy-sensitive information of the monitored person. An attacker may get access to that personal information to compromise the privacy of the monitored person. The attacker may also compromise the data integrity and transfers false or fake information about the monitoring person to the end-user monitoring device. These attacks surfaces for the captured data are growing due to the resource limitation and untrusted software stacks (e.g., Operating system, middleware, or user-specific software) of smart camera devices. These attacks may also possible due to using the open infrastructure (e.g., Internet) and public storage devices (e.g., backup server).

The data captured by smart cameras required data protection and security techniques to provide the authentication, integrity, non-repudiation, freshness and confidentiality. The captured data further needs realtime transmission to a monitoring device in a secure and protected way to fulfill the surveillance requirements (e.g., independent living or automated assistance etc.) for remote monitoring. The lifetime protection and security of data are necessary to realize the advantages and widespread adaptation of the monitoring use-case in such application scenarios (e.g., AAL scenario).

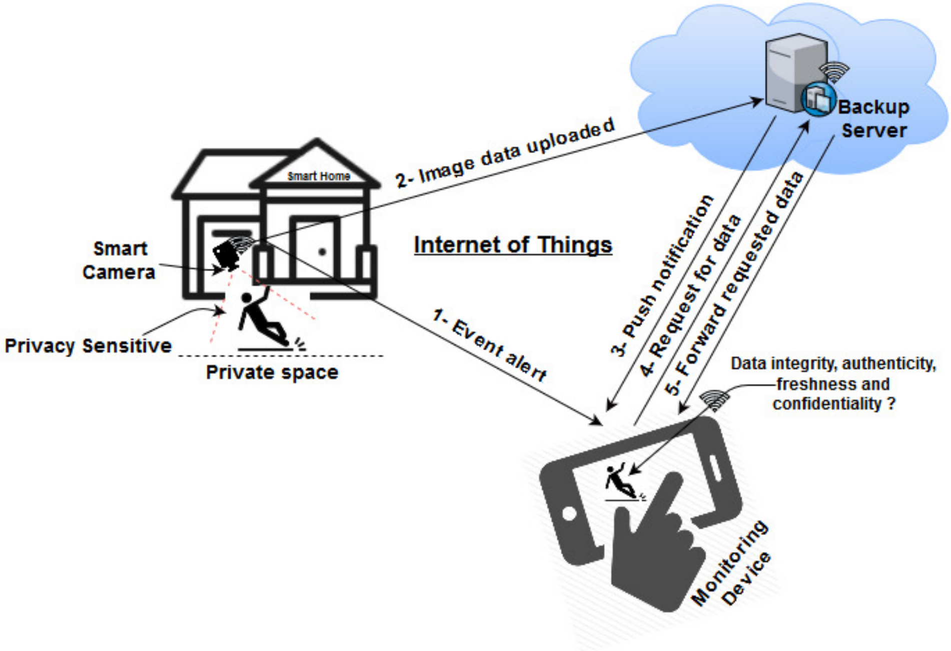

Figure 3.1: Monitoring use-case for ambient assisted living (AAL) IoT scenario: The AAL monitoring scenario consists of a set of distributed smart cameras, a backup server for permanent storage and end-user monitoring device (e.g., caretaker smartphone). All these devices communicating with each other, where the arrows and labels show the flow of data.

3.2 System Architecture

This section presents the proposed system architecture, its integral components and connectivity in a typical IoT environment. The integral components are smart cameras (C_j), a backup server (BS) and monitoring devices (M_h), where j and h represent the identifiers for the smart camera and the monitoring device, respectively. Figure 3.2 shows the internal view of the system, where the smart camera is further divided into a sensing unit and a camera host unit. The internal view describes the functionalities of the system, where the sensing unit captures the image sequence of the target scene and detects RoI through pre-defined low-level video processing algorithms. The RoI data is the important information required for the surveillance of a monitoring place or object (e.g., the image data of the face of a person). The RoI data should be enough to fulfill the specific surveillance requirements (e.g., tracking, identification and detection). The camera host unit consists of different hardware and software stacks e.g., operating system, network stack, system libraries, to run and manage the camera application. The RoIs serve as event data and are transfered to the camera host unit which is responsible for further processing and for transmitting them to the backup server and alerting the monitoring device about the detected events.

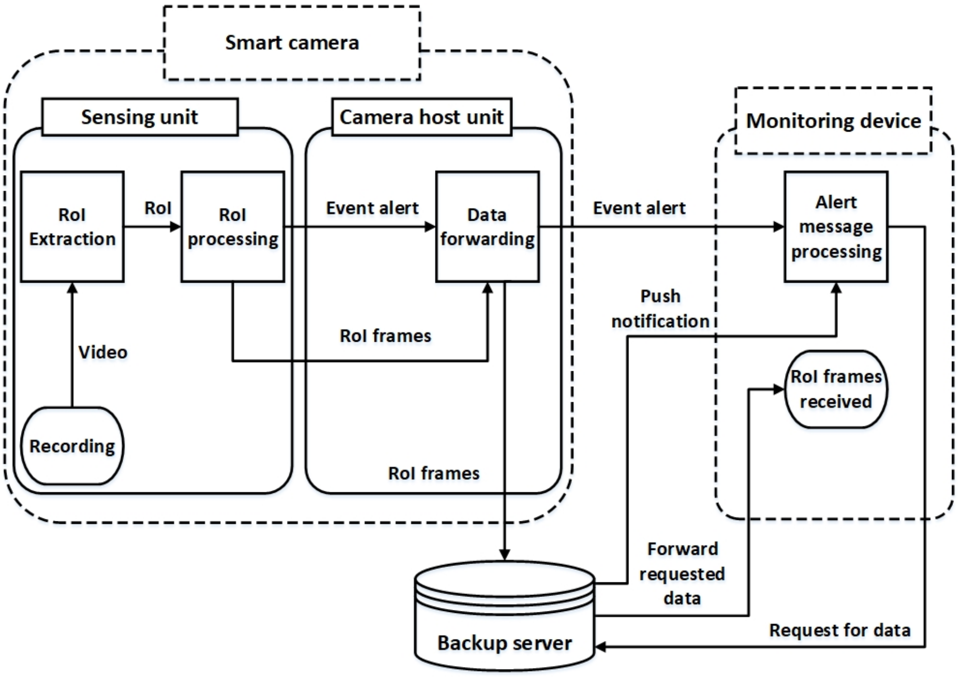

Figure 3.2: System architecture: The system architecture shows the functionalities and complete end-to-end communication in the system. The arrows with labels show the flow of data in the proposed system architecture.

Typically, a smart camera has not sufficient storage for all captured data because of smart camera resource limitations, so a backup server (BS) is used to permanently store the data for the intended monitoring device. The backup server provides authorized access to that stored data for the corresponding monitoring device (h).

The system architecture proposed in this work consists of three different communication scenarios based on the transfer of captured information from smart cameras to monitoring devices. This classification is performed according to the specific requirements of end-user applications. The requirements of end-user application depend on the monitoring space, the number of smart cameras and the number of end-users devices. This thesis presents efficient data protection and security approaches for all these scenarios. In a simple scenario, a single smart camera transfers the captured information to a single dedicated monitoring device. Subsection 3.2.1 introduces such a simple scenario and in the rest of the thesis, it is referred to as single-sender/single-receiver scenario. Subsection 3.2.2 presents the second scenario, in which multiple smart cameras transfer the captured information to a single monitoring device. In the rest of the thesis, it is referred to as the multi-sender/single-receiver scenario. Finally, Subsection 3.2.3 introduces a more complex communication scenario in which multiple smart cameras transfer their captured information to multiple monitoring devices. This scenario is called as multi-sender/multi-receiver scenario in the rest of the thesis.

3.2.1 Single-Sender/Single-Receiver Scenario

A smart camera j can perform surveillance tasks independently and shares the data with a single monitoring device h as shown in Figure 3.3. The smart camera uses a backup server (BS) to permanently stores the captured data for the intended monitoring device. The backup server provides an authorized access to that stored data for the corresponding monitoring device (h). The detailed processing and data flow of this scenario is shown in the Figure 3.2.

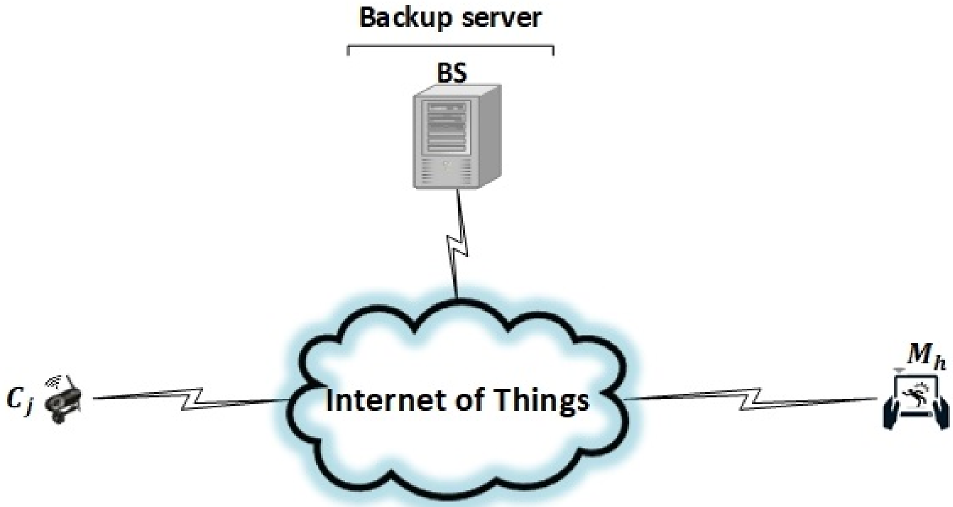

Figure 3.3: Single-Sender/Single-Receiver Scenario: A single smart camera j captures (images/video) data and shares them with a single monitoring device h.

3.2.2 Multi-Sender/Single-Receiver Scenario

In IoT applications, a network of smart cameras is used for the surveillance of a large area. This work proposes a cluster-based network and group the co-located smart cameras into distinct clusters [185] as shown in Figure 3.4. Each cluster has a pre-defined cluster head CH_i where i represents the identifier of the cluster. The cluster head works as a gateway [159] and connects the smart cameras with the rest of the system. Smart cameras use a backup server (BS) to permanently store the aggregate data forwarded by the cluster heads for the intended monitoring device. The backup server provides an authorized access to that stored data for the corresponding monitoring device (h).

This is the extension of the single-sender/single-receiver scenario (Figure 3.3) to a multi-sender/single-receiver communication scenario (Figure 3.4). Here, multiple cameras provide data for detected events and need to secure it for an individual monitoring device. The processing for such scenario can be summarized as follows: (i) onboard detection of predefined events on smart cameras in a cluster, (ii) aggregation of the information on the cluster head, (iii) transfer and storage of the aggregated information on a backup server, (iv) and download of the information by monitoring device stored on the backup server to complete the surveillance procedure.

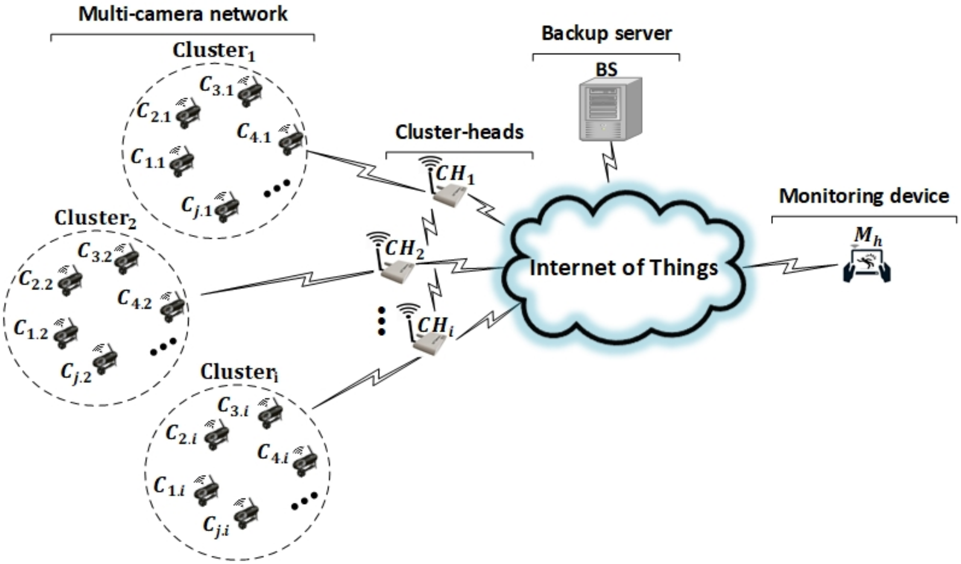

Figure 3.4: Multi-Sender/Single-Receiver Scenario: Multiple smart cameras j capture (images/video) data and share them with a single monitoring device h.

3.2.3 Multi-Sender/Multi-Receiver Scenario

This section generalizes and extends the previous scenarios (Figure 3.3 and Figure 3.4) of smart cameras to a multi-sender/multi-receiver scenario (Figure 3.5) for IoT applications, where a group (cluster) of smart cameras provides secure data of detected events to multiple monitoring devices. A third-party trusted entity KGC is responsible for the partial key generation and public key authentication in this scenario. The KGC initiates the system setup and key sharing, where the shared keys are used for secure communication in the system. The key processing steps for such scenario can be summarized as follows: (i) onboard detection of predefined events on the smart cameras within a cluster, (ii) aggregation of the information on the cluster head, (iii) storage of the aggregated information on a backup server and (iv) download of the information which are already stored on the backup server for the respective multiple monitoring devices to complete the surveillance procedure. The KGC provides a certificateless approach for the public key generation and authentication in this multi-sender/multi-receiver scenario. The public key generation and authentication procedure using KGC for smart cameras and monitoring devices is shown as in Figure 6.2.

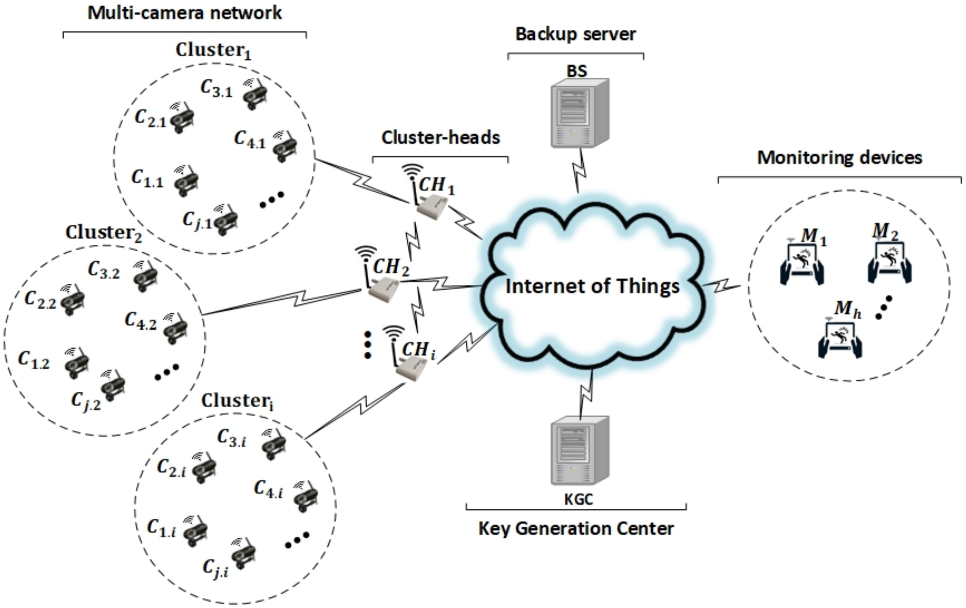

Figure 3.5: Multi-Sender/Multi-Receiver Scenario: Multiple smart cameras j capture (images/video) data and share them with multiple monitoring devices h.

3.3 Threat Model

Smart cameras in IoT applications (e.g., smart home monitoring) have several vulnerabilities [13, 111, 170], which can be exploited by attackers to gain root access to the smart camera nodes and compromise the security and privacy of data. The open infrastructure (e.g., Internet) in IoT applications pose a challenge to mitigate such attacks and to secure the smart cameras from unauthorized access. An attacker may get access to the camera host unit, cluster head, communication channel and backup server in all scenarios of the system architecture (Section 3.2). In a multi-receiver scenario, one monitoring device can observe the data which are intended for the other monitoring devices and can compromise the confidentiality. In video surveillance and monitoring applications, the important assets to protect from unauthorized access are the captured sensitive information (images/videos), the secret keys (e.g., private keys and encryption keys) of smart cameras and the camera node itself. The unauthorized access to the sensitive and personal information can compromise the confidentiality, authenticity, integrity and freshness of data. The identification of these threats is required to avoid such attacks and provide trust and reliability of the shared information in these scenarios. A threat model predicts the information and identification about the possible attacks and vulnerabilities, which are necessary to prevent and countermeasure in the deployment of system architecture. The threat model in our proposed system architecture addresses the following eavesdropping, data modification, impersonation and replay attacks to the information transmitted from the smart camera to the monitoring device.

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.

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 monitoring device. Compromised keys are common reasons for this attack. The attacker can compromise the integrity and alter the data while remain undetected.

Impersonation. In the impersonation attack, an attacker obtains the identity of a smart camera by spoofing the hardware address (MAC address) or by intercepting the data frames on communication channels. The attacker uses that identity and gets unauthorized access to the network. The attacker successfully programmed the compromised identity in their own device and then transmit its own data. The attacker cheats the end-user device by sending wrong information or introducing fake data in the smart camera systems.

Replay Attack. In replay attacks, the attacker gets unauthorized access to the capture data of the smart camera onboard or by intersecting the communication channel. They keep the valid information (image/video) and transmit them repeatedly by injecting them in the communication channel and replacing the live image or video data. In this attack, the attacker intercepting the data frames and delivers outdated information as fresh one to the end-user monitoring device.

3.4 Requirements and Assumptions

This work assumes that each smart camera consists of a trusted sensing unit and camera host unit [167], and hence that the attacker has no access to data on the sensing unit. This trusted sensing applies a fingerprint and attestation for the captured data. The fingerprint provides a unique signing (attestation) key for the of captured data which is used for the generation of signcryption. The data protection techniques implemented in the sensing unit provide reliability and trustworthiness of the captured or sensing data. The protection of the sensing unit is built upon the work [171, 61] and has exclusive access to the raw data (images and videos). The host unit is not explicitly trusted because of the operating system (OS), libraries, middle-ware and other user-specific applications. The camera host unit is responsible for the configuration, management and running of the applications and system libraries. Hence, 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 [111] and VSN [170] scenarios. The DoS attacks on the camera host unit or backup server are not explicitly considered in this work. The threat model discussed in the Section 3.3 identifies those attacks only which compromises the protection and security of the captured data not the software or hardware corruption. The corruption of software or hardware and DoS attacks on the camera host, backup server or monitoring device can disrupt the normal functionality of the proposed system architecture. While assuming the sharing of public parameters and public keys, this work can verify the authenticity of incoming requests by using the public parameters and public keys, which is called public verifiability (a property of the signcryption technique). The public verifiability of data can reduce incoming requests of an attacker and only forward authentic requests. Moreover, this work assumes that the monitoring device is trustworthy and that the private key is securely stored on it. This is also assuming that the backup server provides authorized access to the stored data for authentic monitoring devices.

This work further implements KGC for the complex multi-sender/multi-receiver scenario and proposes that the KGC generates the partial private keys for all components and can verify and authenticate the public keys of all components within the system architecture. The public keys are generated by the smart cameras based on their private and partial private keys and securely share them with the monitoring devices and other components of the system architecture.

3.5 Data Protection and Security Approaches

This work considers all the security threats identified in Section 3.3 and presents secure approaches for each scenario of the system architecture (Section 3.2). First, this work presents signcryption techniques to secure single-sender/single-receiver smart camera scenario. 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. Second, this work uses aggregate-signcryption for multi-sender/single-receiver smart camera scenario and presents a cluster-based secure approach for the system architecture to reduce the risks of attacks in open infrastructure. An aggregate-signcryption efficiently merges the signcryptexts of distinct smart cameras into a single and smaller aggregate-signcryptext. It merges the public key information of the intended monitoring device in a compact form. Then the monitoring device uses that information and verifies the authenticity and integrity of all data in a single step. Aggregate-signcryption does not affect the security of individual signcryptexts. Third, this work presents certificateless multi-receiver aggregate-signcryption techniques with public verifiability, decryption fairness, and forward secrecy to avoid the key escrow problem and guarantee the protection, verification, and exclusive access to the data only by authentic/intended users. In the following, this chapter presents a brief introduction of the EC-based signcryption setup and approaches for the security of each relevant scenario.

Signcryption Setup. This thesis implements signcryption [95] 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 [62]. An EC over the finite field F_q is represented by $E(F_q)$ with a base point $G \in F_q$ of order

n, where G is chosen randomly from the set of points on $E(F_q)$. The parameter q is a prime number specifying the finite field F_q.

Secure Camera for Single-Sender/Single-Receiver Scenario. In the system architecture shown in Figure 3.2, 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). 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 monitoring device and the backup server, respectively. When the server receives new data from the smart camera, it sends a push notification to the monitoring device. As soon as the monitoring 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 is beyond the scope of this work. The approach of secure smart cameras with onboard signcryption is refer as individual-signcryption, and this thesis presents its implementation and evaluation details in Chapter 4 and available online [154].

Secure Camera for Multi-Sender/Single-Receiver Scenario. This subsection introduces lightweight security techniques to protect and secure the sensitive information in the multi-sender/single-receiver scenario of the system architecture (shown in Figure 3.4). This approach extends the individual-signcryption techniques (cp. Chapter 4) for the security of multi-sender/single-receiver scenario. The design goals of this security technique are (i) to reduce the transmission of unnecessary data, (ii) to protect the captured information from unauthorized access throughout its lifetime, and (iii) to prove the authentication and integrity of the information on the intended monitoring devices. The approach of securing cluster-based smart cameras network is refer as aggregate-signcryption, and this thesis presents its implementation and evaluation details in Chapter 5 and also available online [155].

Secure Camera for Multi-Sender/Multi-Receiver Scenario. In this approach, the thesis extends the individual-signcryption and aggregate-signcryption approaches to multi-receiver aggregate-signcryption approach for the security of multi-sender/multi-receiver scenarios. This work added techniques of exclusive protection and decryption fairness for the exclusive access and decryption of the same data by multiple receivers. This work further added forward secrecy to maintain the confidentiality of incoming or past data from smart cameras in the case of compromising of a specific session key by an attacker at any stage. The security requirements for the proposed multi-sender/multi-receiver scenarios can summarize as follows: (i) authentication and sharing of the public keys in advance, (ii) exclusive protection of data on smart cameras for different receivers using symmetric keys (exclusive protection on the sender side), (iii) optimization of aggregate-signcryption to the multi-receiver scenario, (iv) exclusive access to the received signcrypted data by multi-receiver (exclusive access on the receiver side), (v) maintaining decryption fairness, which is the exclusive decryption of data by monitoring devices using their own session keys, (vi) public verifiability of the data by any trusted or untrusted party and (vii) forward secrecy when session keys are compromised by an attacker. This thesis presents the secure approach of multi-receiver aggregate-signcryption with decryption fairness in Chapter 5 and available online at [153].

Chapter 4

Onboard Signcryption

Overview

This chapter discusses the EC-based signcryption for the security and protection of data onboard the smart camera IoT applications having single-sender/single-receiver communication scenario. In a single-sender/single-receiver communication scenario, a stand-alone smart camera performs the surveillance tasks (event detection, image or video capturing, object tracking, data processing and transmission etc.) of a target location independently and shares the collected information with an end-user monitoring device. The primary work of this thesis implements the EC-based signcryption onboard for the stand-alone smart cameras which is also referred to as individual-signcryption in the rest of the thesis. Individual-signcryption provides the functionality of signing and encryption in a single step implementation with reduced computation and communication overheads as compare to sign-then-encryption implementation procedure. This chapter presents the EC-based signcryption setup, signcryption and unsigncryption algorithms and the security analysis for stand-alone smart camera. This chapter evaluates the efficiency of signcryption procedure in term of computation and measures the running time of onboard signcryption procedure by varying the EC-key and data sizes. Section 4.1 presents individual-signcryption and explains the key generation procedure, signcryption and unsigncryption algorithms. Section 4.2 presents security analysis to countermeasure the possible attacks in the proposed scenario of the system architecture. Finally, Section 4.3 explains the experimental setup and concludes the chapter with a discussion of the results.

4.1 Individual-Signcryption

This section presents the individual-signcryption and covers the key generation, signcryption and unsigncryption procedures. The algorithms for key generation, signcryption and unsigncryption are based on the setup proposed in the overall approach of Chapter 3 (Section 3.5).

4.1.1 Key Generation

A trusted entity called PKG generates and shares the public parameters of the system, which are used by the participating devices to define their own security setup. The EC base point G, finite field F_q, prime number q [62] and the required characteristics (e.g., the type and length of keys) are included as public parameters in the preliminary setup [21]. These security parameters are fixed and generated in advance by the PKG during the deployment phase. The PKG generates the private keys for devices (e.g., smart camera and monitoring device) in a system architecture. The private keys based on a mathematical operation taking the identity of the device and the master secret key of PKG as input. Assume that the PKG generates the private key Pr and the public key Pu using EC for the smart camera and other devices in the proposed system architecture. The private key is randomly chosen from a set of large prime numbers. The public keys are derived from the points of elliptic curve on the basis of the chosen private key, e.g., $Pu = Pr{\cdot}G$ is called elliptic-curve-based discrete logarithm problem (ECDLP). The PKG generates the camera's private key Pr_j and the public key Pu_j. It also generates the private key Pr_h and the public key Pu_h of the monitoring device. These keys are distributed by a key distribution center (KDC) in a secure way during initialization of the system or joining of a new device.

4.1.2 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 smart camera sensing unit captures video data upon some event detection or upon the request of end-user devices. The sensing unit extracts RoI frames from the captured video or images and generates an alert message to inform the end-user. The sensing unit applies signcryption on the RoI of captured video or image frames and on the alert message. The signcryption algorithm chooses a prime number v where $v \in \{2, 3, ...n - 1\}$. In signcryption algorithm the Equ. 4.1 generates k_1 by hashing the EC-point computed from the EC-point multiplication of v and G (EC generator or base point). The Equ. 4.2 computes the encryption

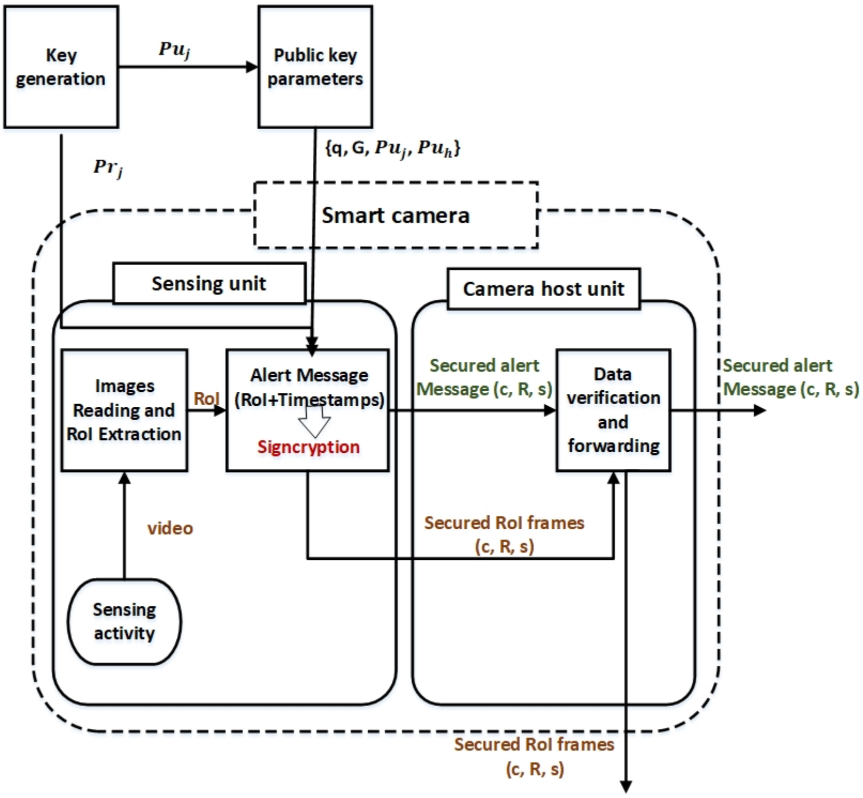
Key generation
Pu_j
Public key parameters
$\{q, G, Pu_j, Pu_h\}$
Pr_j
Smart camera
Sensing unit
Camera host unit
Images Reading and RoI Extraction
RoI
Alert Message (RoI+Timestamps)
Signcryption
Secured alert Message (c, R, s)
Data verification and forwarding
Secured alert Message (c, R, s)
video
Secured RoI frames (c, R, s)
Sensing activity
Secured RoI frames (c, R, s)

Figure 4.1: Signcryption procedure onboard the smart camera.

key K_{enc} by hashing EC-point computed from the resulted value of the point computation of v and the public key of monitoring device Pu_h. The K_{enc} is used in Equ. 4.3 to generate the ciphertext c. The k_1 is further concatenated with the ciphertext c and computed r by hashing that concatenated value as shown in Equ. 4.4. The r value is further used for the computation of R value as in Equ. 4.6. The R value is sent as part of the signcrypted data which provides the proof of integrity and authentication on the receiver side (e.g., on monitoring device). The signcrypted alert messages and signcrypted frames represented by green and brown color in the form of (c, R, s) are transferred to the camera host unit as Figure 4.1 shows.

$$
\begin{aligned}
k_1 &= hash(v{\cdot}G) && (4.1)\\
K_{enc} &= hash(v{\cdot}Pu_h) && (4.2)\\
c &= Enc_{K_{enc}}(RoI_{frame}) && (4.3)\\
r &= hash(c, k_1) && (4.4)\\
s &= \frac{v}{(r + Pr_j)}\, mod\, q && (4.5)\\
R &= (r{\cdot}G) && (4.6)\\
Signcryption\ output &= (c, R, s) && (4.7)
\end{aligned}
$$

Then the camera host verifies the authenticity of the signcrypted data with the public key of the smart camera and considers it as authentic if $r{\cdot}G = R$, 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 successful verification the camera host forwards the secured alert message and frames to the monitoring device and the backup server, respectively.

4.1.3 Unsigncryption Algorithm

When the monitoring device receives an alert message and encrypted video frames, it performs the following un-signcryption algorithm as shown in Figure 4.2.

$$
\begin{aligned}
k_1 &= hash(s(R + Pu_j)) && (4.8)\\
r &= hash(c, k_1) && (4.9)\\
K_{dec} &= hash(Pr_h(s(R + Pu_j))) && (4.10)\\
RoI_{frame} &= Dec_{K_{dec}}(c) && (4.11)\\
r{\cdot}G &= R && (4.12)
\end{aligned}
$$

4.2 Security Analysis

In this section the security analysis of the signcryption scheme (Section 4.1) with specific attention to the single-sender/single-receiver scenario of the system architecture (Section 3.2.1) is presented, in order to countermeasure the

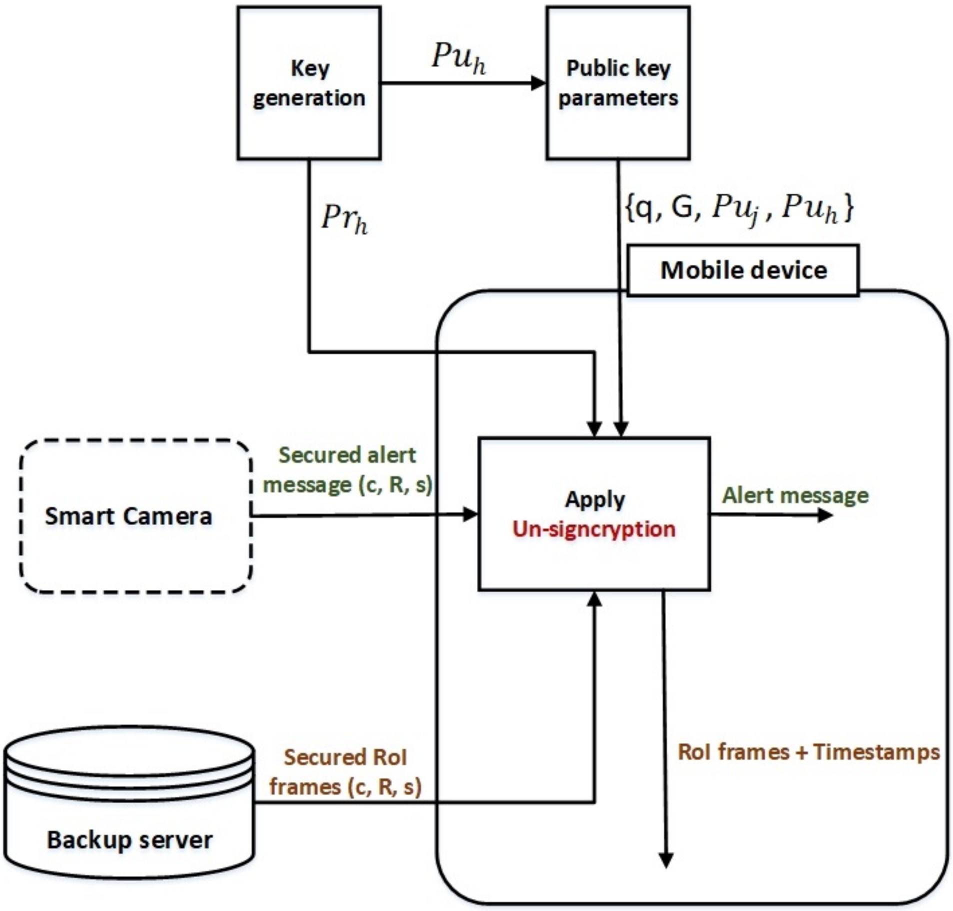

Figure 4.2: Un-signcryption procedure onboard the smart camera.

attacks identified in the threat model (Section 3.3). The basic security goals of these countermeasures are confidentiality, integrity, authenticity, freshness of the data processed by the smart camera. The security of signcryption is based on the assumption of computational hardness of ECDLP [181].

Confidentiality. Confidentiality of image or video frames is provided by AES encryption using a session key (encryption key) K_{enc} during the signcryption process. The K_{enc} is derived by using a secret key v and the public key of the monitoring device Pu_h (cp. Equ. (4.2)). In this case, the attacker needs to know v to derive K_{enc}. To guess v corresponds to solving the ECDLP. Another possibility for an attacker is to solve Equ. (4.10) to derive K_{dec}, but in this case the attacker only knows the public key of the monitoring device Pu_h but not the private key of monitoring device Pr_h. To derive the private key of the monitoring device attacker needs to solve the ECDLP again. It means that the encryption key is secure from both the sensing unit and the monitoring device perspective to the attacker.

Integrity. The signcryption technique provides proof for the data integrity (e.g., on monitoring device). This proof is possible by re-computation of the k_1 value as in Equ 4.8. The k_1 is the hash value which requires Pu_j (public key of smart camera) and the signcryptext parts (s and R) as input. The monitoring device recomputes the value of r by hashing the concatenated value of c (encrypted part of the received signcrypted data) with k_1 as shown in Equ. (4.4). Anyone can check the integrity of the encrypted data using k_1 derived in Equ. 4.8. If an attacker modifies the encrypted data c to c', the change will be detected on the monitoring device because of collision resistance of the hash function. The modified value will give a false result of the Equ. 4.12. 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.

Authentication. It is important to know the identity of the smart camera which is claiming the capturing of the image or video data. This identity proof is required on the monitoring device before applying the decryption procedure of the data. The r value is recomputed as shown in Equ. 4.9 for the integrity proof. This recomputed value of r and G is then compared with the received R value as shown in Equ 4.12. If both side of the Equ 4.12 results in equal value then the data will be authentic otherwise not. The authentication of data provides proof of the known identity of the smart camera. The equal value of recomputed k_1 on monitoring side is equal to the k_1 value on smart camera side prove the identity because of the following proofs e.g.,
$hash(s(R + Pu_j)) = hash(v{\cdot}G) = (k_1)$ from Equs. (4.1) and (4.8).

Freshness. Timestamping provides freshness of data and prevents replay attacks. This work assumes that image or video frames are securely timestamped before signcryption. The signcryption of timestamped data provides proof of freshness of the data. The monitoring device verifies the validity and timestamp after the successful processing of unsigncryption of the protected timestamped data.

4.3 Experimental Setup and Results

This section evaluates the efficiency of EC-based signcryption in term of computation and communication for the data protection and security onboard the smart camera. The implementation of the proposed EC-based signcryption is performed on a Raspberry Pi-3 which has an 1.2 GHz ARMv8 CPU

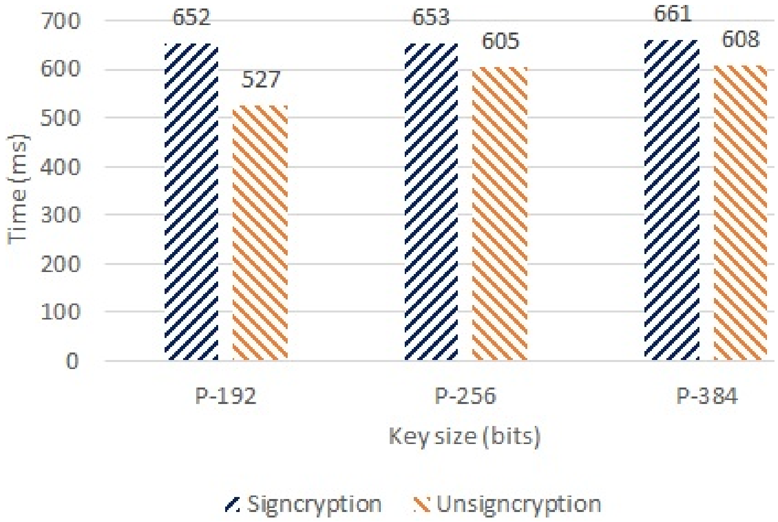

Figure 4.3: Running time of signcryption and unsigncryption with different EC keys for an 480 × 320 image with a size of 105 kB.

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, this work integrated signcryption and unsigncryption in a single Java package and measured the running times. This work investigated the running times for protecting single RoI image in two different experiments. In the first experiment (as shown in Figure 4.3), this work 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 Figure 4.4), this work 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. (4.2) and (4.3)). Thus, although keys with variable bit lengths are provided, encryption is always performed with the 256-bit key (K_{enc}).

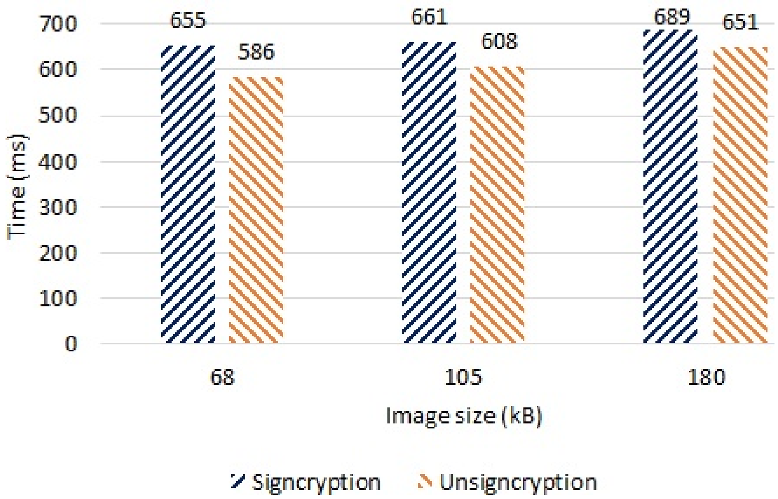

Figure 4.4: Running time of signcryption and unsigncryption with different image sizes using an EC P-384 key

4.3.1 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 [96] or RSA-based digital signature [171] with AES encryption. It was demonstrated [181] 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 the encryption or decryption part influences the running time. A hardware accelerator for the hash function, AES and EC on the smart camera can improve the computational efficiency.

Chapter 5

Multi-Sender Aggregate-Signcryption

Overview

This chapter presents the protection and security techniques to secure the data of multiple smart cameras for a single monitoring device in a multi-sender/single-receiver communication scenario. The individual-signcryption (Chapter 4) protects and secures the captured data of stand-alone smart cameras in a single-sender/single-receiver scenario with reduced computation and communication overheads. The bottleneck of individual-signcryption in multi-sender/single-receiver communication scenario is the sequential verification of signcrypted data of each individual smart camera on a single monitoring device. This chapter presents aggregate-signcryption techniques for the cluster-based multiple smart camera networks. A cluster consists of a group of cameras and a cluster head. The cluster head is a special node which aggregates the individual-signcryptexts of smart cameras of the cluster. The cluster head sends the aggregated data to a monitoring device. The monitoring device verifies the integrity and authenticity of aggregate-signcryptext. This verification provides the integrity and authenticity for all individual smart cameras of that specific cluster in a single step. Section 5.1 introduces the aggregate-signcryption approach and its deployment steps. Section 5.2 presents the deployment phase and the setup of aggregate-signcryption. Section 5.3 presents the operational procedure of aggregate-signcryption and unsigncryption algorithms. Section 5.4 discusses the security analysis of the aggregate-signcryption approach. Finally, Section 5.5 presents the experimental setup and results.

In the following, this chapter presents an overview of the lightweight security approaches of aggregate-signcryption as well as its deployment and operational phases in the system architecture.

5.1 Aggregate-Signcryption

This section presents an aggregate-signcryption, in order to merge the signcrypted information from individual smart cameras of a cluster and to reduce the transmission of redundant information. The aggregate-signcryption provides the proof of integrity and authenticity for all the smart cameras of cluster head in a single step with reduced computation overheads as compared to sequential verification of individual signcryptexts of smart cameras on the monitoring device.

Key Generation. This work assumes a KGC [9], a trusted entity in the system to generate the partial private keys for all devices of the system. The KGC securely shares the partial private keys with the respective devices, and then the devices generate their full private and public keys. This work assumes that all devices keep the private keys secret and share their public keys with each other in the system.

Local Analysis and Onboard Signcryption. The smart cameras perform local event detection and then extract the RoI. The smart camera applies EC-based signcryption to protect the selected data on the sensing unit [154]. The protected information is forwarded to the camera host unit, which can verify the integrity and authenticity of the data and transfers it to the cluster head for performing aggregate-signcryption.

Aggregate-Signcryption on Cluster Head. The cluster head of each cluster verifies and aggregates the individual signcryptexts of the received data from the detected event. The algorithm of aggregate-signcryption is defined in Section 5.3.2. There is no need to share private keys with the cluster head but only the identities and public keys of the corresponding smart cameras and the receiving monitoring device is needed as input for the aggregate-signcryption.

Permanent Backup of Data and Accesses Authorization. The cluster head forwards the aggregate-signcryptexts to a backup server for permanent storage and alerts the monitoring device. The monitoring device accesses the relevant aggregated-signcryptext and then performs the verification and unsigncryption on it.

5.2 Deployment Phase

In the deployment phase, the system initiates setup of the entities and shares the identities along with the associated public keys and other state information.

A KGC generates and shares the public parameters of the system, which are used by the participating devices to define their own security setup. The EC base point G, finite field F_q, prime number q [62] and the required characteristics (e.g., the type and length of keys) are included as public parameters in the preliminary setup [21]. These security parameters are fixed and generated in advance by the KGC during the deployment phase.

Setup Initialization. The KGC runs the setup algorithm and takes $k \in \mathbb{Z}_q^+$ as input (k specifies the bit length) to generate the partial private keys and public parameters [9, 95]. The EC parameters are based on the chosen signcryption setup proposed in the Section 3.5. The key generation procedures for aggregate-signcryption is not completely based on a third party trusted entity (e.g., PKG), however aggregate-signcryption uses the KGC for partial private key generation for each device (e.g., smart camera) upon a partial key request. Then each device uses that partial private key and further computes its full private key. It is assumed that each device generates its full private key Pr and the public key Pu on the basis of the partial private key. For example, the smart camera j generates its private key as Pr_j and the public key Pu_j. The monitoring device (h) also generates its private key Pr_h and the public key Pu_h. Each device keeps the private key secret and shares the public key during the initialization of the system or joining of a new device.

5.3 Operational Phase

In the operational phase each smart camera initiates the signcryption process and generates a session key $K_{enc(j)}$ by using the public key Pu_h of a monitoring device. The smart camera uses its private key Pr_j for the signature part, while the session key for the encryption part and performs signcryption on the captured data as following.

5.3.1 Signcryption by Smart Camera

Let's suppose that a smart camera C_j of cluster i detects an event and starts the signcryption procedure. Each smart camera j selects the internal state information ω (e.g., firmware version and timestamp of the health status) to confirm the secure execution of the program and then performs the signcryption by executing the following steps:

- selection of a prime number $v_j \in \mathbb{Z}_q^*$,

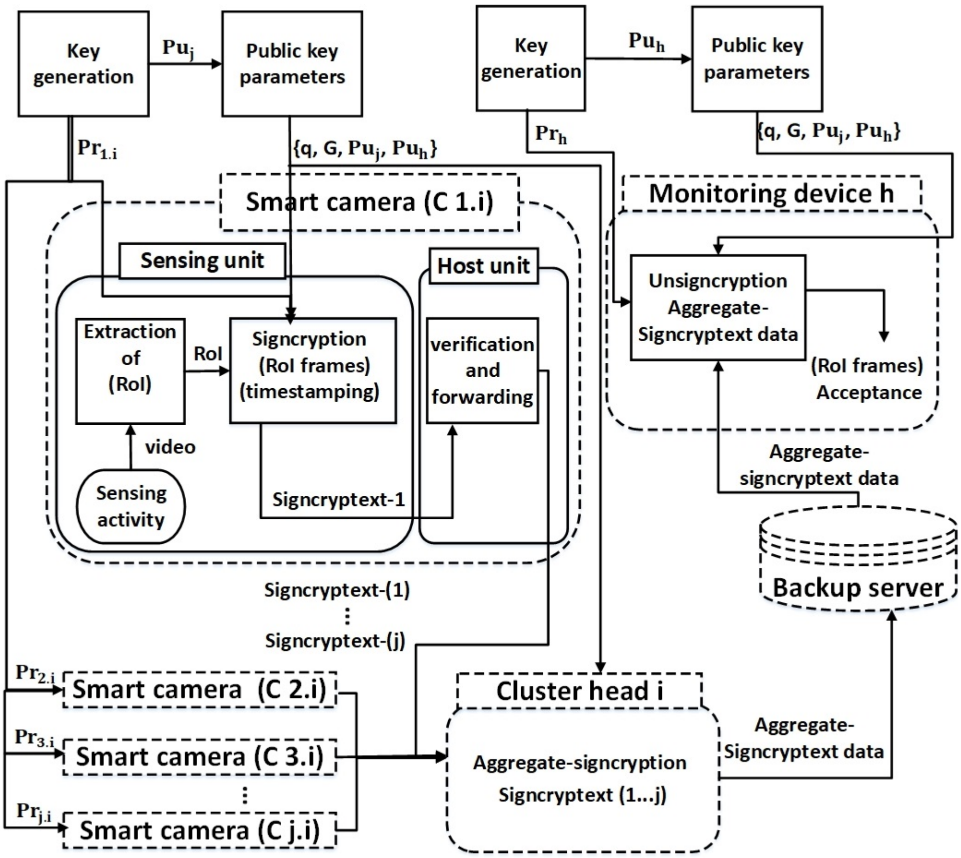

Figure 5.1: Processing flow of the cluster-based aggregate-signcryption.

- computation of $k_{1(j)} = hash(v_j{\cdot}G)$,
- generation of the session key as $K_{enc(j)} = hash(v_j.Pu_h)$,
- encryption of the RoI of the video frames.

$$c_j = enc_{K_{enc(j)}}(RoI_{frames(f)})_j \tag{5.1}$$

$$r_j = hash(c_j, k_{1(j)}) \tag{5.2}$$

$$s_j = \frac{v_j}{(r_j + Pr_j)}\, mod\, q \tag{5.3}$$

$$R_j = (r_j{\cdot}G) \tag{5.4}$$

$$Signcryptext = (c_j, R_j, s_j) \tag{5.5}$$

Each smart camera forwards its signcryptext packet (c_j, R_j, s_j) to the cluster head i.

5.3.2 Aggregate-Signcryption Algorithm

The cluster head i performs the aggregation of the individual-signcryptexts received from the cameras. The aggregate-signcryption takes the public keys of the smart cameras Pu_j, the public key of monitoring device Pu_h and corresponding signcryptexts. The cluster head first verifies the individual signcryptexts and then generates the aggregate-signcryptext as following:

- computing $S = \sum_{j=1}^{n} s_j$ and parse the c_j and R_j in a sequential order.
- merging signatures and encrypted data $(c_1 \cdots c_j, R_1 \cdots R_j, S)$ as aggregate-signcryptext.

5.3.3 Aggregate-Unsigncryption Algorithm

Prior to the decryption of aggregated data, the monitoring device first verifies the acceptance (authentication and integrity) of the aggregate-signcryptext data by using its own private key Pr_h, the associated public keys of the smart cameras Pu_j and the received aggregate-signcryptext. In case of success, the output of the unsigncryption algorithm is the individual-signcryptexts (c_j, R_j, s_j). This single step verification of aggregated data is true for all individual-signcryptexts and there is no need to run the acceptance procedure individually. The monitoring device needs the individual session keys of the smart cameras to proceed with the decryption of c_j. It starts recovering the session keys $K_{dec(j)} = hash(Pr_h(s_j(R_j + Pu_j)))$ and then performs the decryption to get the RoI frames, e.g., $(RoI_{frames(f)})_j = dec_{K_{dec(j)}}(c_j)$.
Figure 5.1 shows the processing flow of the signcryption, aggregation and unsigncryption procedure for multi-sender/single-receiver communication scenario (cp. definition in Chapter 3). The correctness of the scheme to recover $K_{dec(j)}$ on the motoring device is based on the following reasoning:
$hash(Pr_h(s_j(R_j + Pu_j))) = hash(Pr_h(s_j{\cdot}R_j + s_j{\cdot}Pu_j)) = hash(Pr_h(v_j{\cdot}G)) = hash(v_j(Pr_h{\cdot}G)) = hash(v_j{\cdot}Pu_h) = K_{enc(j)}$

5.4 Security Analysis

The security analysis of the aggregate-signcryption scheme with specific attention to the system architecture can be summarized as follows: The basic security goals are confidentiality, integrity, authenticity and freshness of the image or video data. The security of signcryption is based on the assumption of the computational hardness of ECDLP [181].

Confidentiality. Confidentiality is provided by AES encryption using a session key $K_{enc(j)}$ during the signcryption process. The guessing of $K_{enc(j)}$ by attackers corresponds to solving the ECDLP.

Integrity. The sensing unit of the smart camera processes the signcryption part r_j by hashing the encrypted data c_j with $k_{1(j)}$ as in Equ. (5.2). If an attacker modifies the encrypted data c_j to c'_j, the change will be detected on the monitoring device because of the collision resistance of the hash function.

Authentication. The signcryption technique provides the authentication and prove the authenticity of data e.g.,
if $hash(s_j(R_j + Pu_j)) = hash(v_j{\cdot}G) = (k_{1(j)})$.
The correctness of the scheme to recover $k_{1(j)}$ on the monitoring device is based on the following reasoning,
$s_j(R_j + Pu_j) = s_j{\cdot}R_j + s_j{\cdot}Pu_j = (\frac{v_j}{(r_j+Pr_j)})R + (\frac{v_j}{(r_j+Pr_j)})Pu_j = (\frac{v_j}{(r_j+Pr_j)})r_j{\cdot}G + (\frac{v_j}{(r_j+Pr_j)})Pr_j{\cdot}G = \frac{v_j{\cdot}G(r_j+Pr_j)}{(r_j+Pr_j)} = v_j{\cdot}G = k_{1(j)}$

Freshness. Image or video frames are timestamped before initiating the signcryption procedure. Hence, the signcryption protects the timestamped image or video frame. The monitoring device verifies the validity image or video frames after the processing of unsigncryption. The authentication and integrity proofs are automatically applies to the timestamped verification and validity.

5.5 Experimental Evaluation

In this section, the thesis compares the computational and communication overhead for individual- and aggregate-signcryption. A complete prototype of the scenario of the system architecture (Figure 3.4) has been implemented and can be summarized as follows:

Raspberry Pi 3 serve as platforms for the smart cameras. This work uses the JRPiCam [47] Java library for image capturing and processing. Each image has a pre-defined QVGA resolution of 320×240 pixels. The open source library BouncyCastle [3] is used as cryptographic service provider (CSP) with the Java cryptography extension (JCE) and the Java cryptography architecture (JCA) as interface. Signcryption is implemented using the EC-finite field of P-384 and a 256 bit AES key. The cluster head is implemented on a standard laptop (Intel core i5 with 2.6 GHz and 8 GB RAM) running Windows 10. This work used another laptop as prototype for the monitoring device. All platforms are connected via WiFi and data transfer is realized via sockets.

5.5.1 Experimental Results

In the first experiment, this work measured and evaluated the computational and communication overheads of individual-signcryption and aggregate-signcryption by securing 15 images (total size of 74.854 kB data) on each Raspberry Pi device. First, this work varied the number of devices for initiating the signcryption at the same time and measured the total computation and communication overhead of signcryption (on Raspberry Pi 3) and unsigncryption (on the monitoring device). Second, this work measured the total computation and communication overhead for aggregate-signcryption (on Raspberry Pi 3 and on the laptop used as cluster head) and aggregate-unsigncryption (on another laptop used as monitoring device). Table 5.1 shows the comparison of individual and aggregate-signcryption with varying number of smart cameras.

In the second experiment, this work varied the number of images and thus the data size and measured the signcryption time (on Raspberry Pi 3) and unsigncryption time (on monitoring device). This work performed this experiment in a cluster of five cameras where each camera secured a different number of images. Table 5.2 shows the measured runtimes for signcryption and unsigncryption, respectively. The total time for individual-signcryption can be determined for the second experiment as follows: Signcryption is executed in parallel on the cameras, thus the maximum runtime (760 ms) is the limiting factor for this step. Unsigncryption has to be performed sequentially on the monitoring device and can be estimated by the sum of the unsigncryption times (1502 ms) resulting in a total time of 2262 ms. For aggregate-signcryption, the total time is given by the maximum signcryption time (760 ms), the aggregate-signcryption time (349 ms) and the aggregate unsigncryption time (634 ms) which sums up to 1743 ms resulting in a performance ratio of 77%. Table 5.2 also shows that the signcryption time only slightly increases with increasing data size. This effect is because the intensive EC-point computations needs to be executed at the beginning of the signcryption process and only the encryption algorithm is dependent on the data size.

As depicted in Table 5.1, aggregate-signcryption shows a moderate increase of the runtime with increasing number of cluster cameras. This additional effort of aggregate-signcryption is clearly compensated by the signification reduction of unsigncryption time, in particular with larger numbers of cluster cameras.

Table 5.1 also shows that the ciphertext part c_j of each signcryptext packet (c_j, R_j, s_j) has the same size of 74.854 kB (size of 15 images) for the individual-signcryption and aggregate-signcryption, while the signature part varies with the number of cameras in the cluster. As this work uses the finite field P-384, so the signature part (s_j) in Equ. (5.3) of signcryption scheme results in 48 Bytes. The r_j in Equ. (5.2) has 48 Bytes because of the keyed hash function

Table 5.1: Comparison between individual signcryption vs aggregate-signcryption. Legends: (IS: individual-signcryption, AS: aggregate-signcryption, PR: performance ratio, ST: signcryption time (on camera nodes), UST: unsigncryption time (on monitoring device), TT: total time, NT: number of transfers, CD: ciphertext data, SD: signature data, AST: aggregate-signcryption time (on cluster head), AUST: aggregate-unsigncryption time (on monitoring device))

		IS						**AS**						**PR**	
		ST(ms)	UST (ms)	TT (ms)	NT	CD (kB)	SD (Bytes)	ST (ms)	AST (ms)	AUST (ms)	TT (ms)	NT	SD (Bytes)	Computing	Signature data
No. cluster members	1	741	358	1099	1	74.854	72	741	138	358	1237	2	72	112.5%	100%
	2	741	716	1457	2	149.708	144	741	264	388	1393	3	96	95.6%	66.6%
	3	741	1074	1815	3	224.562	216	741	280	453	1474	4	120	81.2%	55.5%
	4	741	1432	2173	4	299.416	288	741	325	581	1647	5	144	75.8%	50%
	5	741	1790	2531	5	374.270	360	741	389	700	1830	6	168	72.3%	46.6%

of SHA-384 and it is further used for the computation of the R_j part using the finite field P-384 as shown in Equ. (5.4) of the signcryption algorithm which also results in 48 Bytes. However, this work uses point compression [76] for the R_j part which reduces the size to the half of its length (24 Bytes). Hence the total extra overhead per individual signcryptext of R_j and s_j results in 72 Bytes. In the case of the individual-signcryption each signcryptext carry 72 Bytes of extra data, while in the case of aggregate-signcryption the s_j part is merged into S using the finite field of P-384, which results in 48 Bytes for the aggregated packet and decreases the extra communication overhead.

Table 5.2: Runtime of signcryption (on camera node) and unsigncryption (on monitoring device) for different number of images. Legend: ST: signcryption time, UST: unsigncryption time

Camera ID	**No. images (size in kB)**	**ST (ms)**	**UST (ms)**
1	1 (5.173)	636	221
2	5 (25.077)	677	242
3	10 (49.99)	721	313
4	15 (74.854)	741	358
5	20 (99.717)	760	368

Chapter 6

Multi-Receiver Aggregate-Signcryption

Overview

In this chapter, the thesis presents lightweight security techniques for the protection and security of sensitive information captured by a smart camera in multi-sender/multi-receiver scenario of the proposed system architecture (Section 3.2). The aggregate-signcryption techniques (cp. Chapter 5) provides efficient data protection and security as compare to individual-signcryption (cp. Chapter 4), however, in the case of multi-receiver scenario the aggregate-signcryption requires to repeat its process for each receiver. This chapter generalizes the aggregate-signcryption for multi-receiver with decryption fairness (the possibility of exclusive access by each receiver to the same aggregate-signcryptext data) and presents a certificateless multi-receiver aggregate-signcryption. The design goals of the security techniques are mainly derived from the considered case studies and can be summarized as: (i) to allow only authentic requests, (ii) to reduce transmission of unnecessary data, (iii) to protect the captured information from unauthorized access throughout its lifetime, and (iv) to prove the authentication and integrity of the information on the intended monitoring devices. The scheme provides data protection and security with decryption fairness for multiple monitoring devices in the proposed system architecture. The proposed techniques can also be used to provide data protection and security for related smart camera applications such as intelligent surveillance (e.g., [143, 12]) or safety monitoring (e.g., the automatic detection of cracks in buildings, bridges and subways tunnels [188]). The rest of the chapter is organized as follows: Section 6.1 introduces the multi-receiver aggregate-signcryption with decryption fairness. Section 6.2 and Section 6.3

present the deployment and operational phases respectively. Section 6.4 discusses security analysis. Section 6.5 evaluates security approaches.

6.1 Multi-Receiver Aggregate-Signcryption Approach

This section presents the security technique of multi-receiver aggregate-signcryption with the decryption fairness to efficiently protect sensitive data onboard the cameras and secures the data transfer from multiple cameras to multiple monitoring devices. This chapter implements these security techniques in two phases (deployment and operational). The deployment phase performs the key generation and key sharing with authentication for smart cameras and monitoring devices. In this phase the KGC defines the system setup, chooses the public parameters, and generates the partial keys for all participating devices (Figure 6.2). The full private and public keys are defined by the smart cameras and monitoring devices themselves to avoid the key escrow problem.

In the operational phase, the smart camera uses its private key, the public keys of receiving devices and the public parameters which are already defined by the KGC to execute the signcryption (Figure 6.1). This work adopts the multi-receiver encryption approach [165, 166] to perform the signcryption procedure. This work then applies aggregation on the cluster head to merge the signcrypted data into a single compact packet. The aggregate-signcryptext data is then sent to the backup server, from where the monitoring device can access and download it. The monitoring devices first check the authenticity and verification of data and then proceed with the decryption by using their relevant decryption keys. The decryption keys are extracted from the received aggregate-signcryptext packet by each monitoring device exclusively.

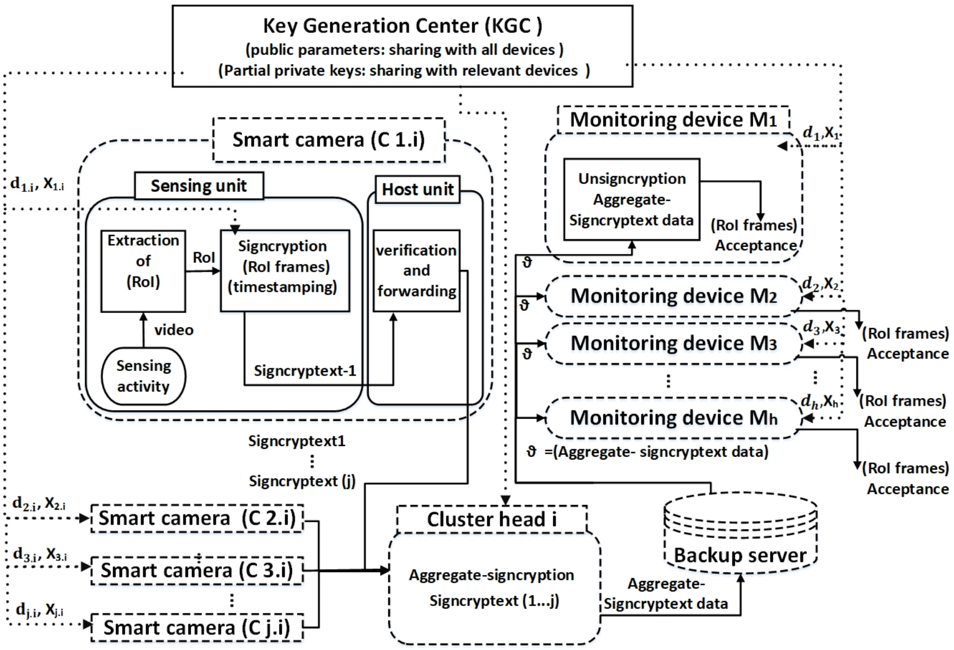

Figure 6.1: Operational phase depicting the processing flow of aggregate-signcryption for smart cameras in cluster i sending protected data to multiple monitoring devices (multi-receiver). The left side shows multi-camera (multi-sender) and the right side shows the monitoring devices (multi-receiver). The distribution of keys and public parameters is shown by dotted lines, and the transfer of actual data is shown by solid lines, where d and X represent the partial keys and ϑ represents the aggregate-signcryptext data for multi-receiver (monitoring devices).

6.2 Deployment Phase

In the deployment phase, the KGC defines the system architecture and security parameters (e.g., EC type, keys length and parameters) in advance, which reduce the load on the resource constrained devices during the operational phase. The KGC is also responsible for the generation of partial private keys for all participating devices. In the deployment phase, the smart cameras are grouped into distinct clusters, and the numbers of relevant monitoring devices for each cluster are defined. The KGC initializes the system setup and generates the partial private keys for all participating devices on request, where each device further defines their full public key which is partially depending on the partial private key which is already received from the KGC. The processing steps of key generation and distribution are shown in Figure 6.2 and explained in the following section.

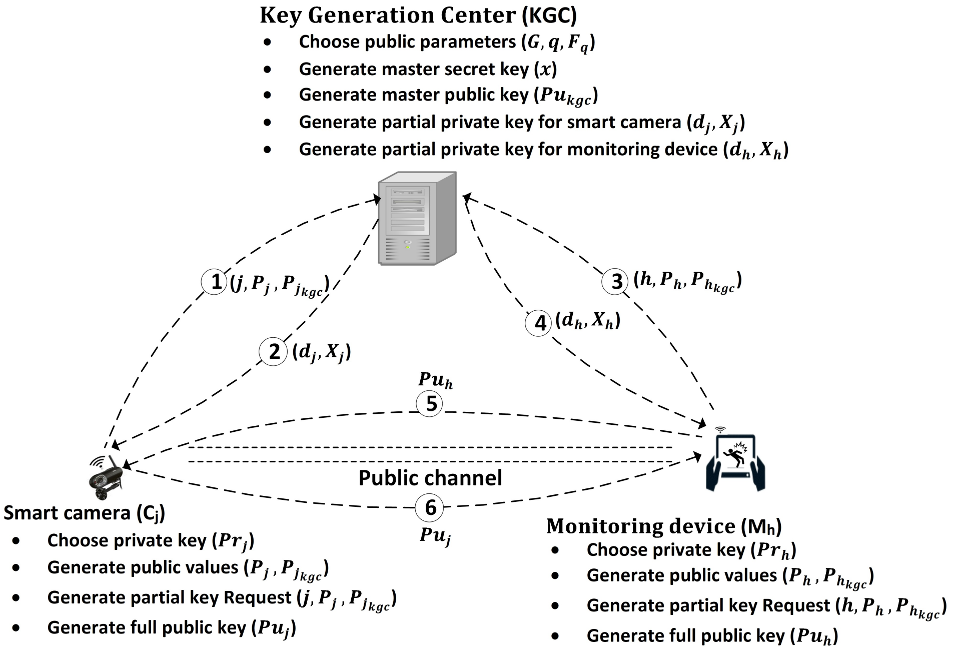
Key Generation Center (KGC)
Choose public parameters (G, q, F_q)
Generate master secret key (x)
Generate master public key (Pu_{kgc})
Generate partial private key for smart camera (d_j, X_j)
Generate partial private key for monitoring device (d_h, X_h)
1 $(j, P_j, P_{j_{kgc}})$
2 (d_j, X_j)
3 $(h, P_h, P_{h_{kgc}})$
4 (d_h, X_h)
5 Pu_h
Public channel
6 Pu_j
Smart camera (C_j)
Choose private key (Pr_j)
Generate public values $(P_j, P_{j_{kgc}})$
Generate partial key Request $(j, P_j, P_{j_{kgc}})$
Generate full public key (Pu_j)
Monitoring device (M_h)
Choose private key (Pr_h)
Generate public values $(P_h, P_{h_{kgc}})$
Generate partial key Request $(h, P_h, P_{h_{kgc}})$
Generate full public key (Pu_h)

Figure 6.2: Key generation and distribution in the deployment phase. First, the KGC chooses public parameters and the master secret key and then generates the master public key on the bases of a chosen private key. The smart camera and monitoring device choose their private keys and generate their respective public values. They share their public values with KGC in steps 1 and 3 to request partial private keys. They receive the requested relevant partial private keys in steps 2 and 4 from the KGC, respectively. The smart cameras and monitoring devices generate their full public keys based on their relevant private and partial private keys and share them with each other through a public channel in steps 5 and 6.

Preliminary Setup of Security Parameters. A KGC generates and shares the public parameters of the system, which are used by the participating devices to define their own security setup. The EC base point G, finite field F_q, prime number q [62] and the required characteristics (e.g., the type and length of keys) are included as public parameters in the preliminary setup [21]. These security parameters are fixed and generated in advance by the KGC during the deployment phase. The KGC shares these security parameters with all participating devices of the system for further use in the operational phase. The KGC is assumed to be secure from DoS attacks and will be providing the authorize access to authentic devices only (e.g., to known smart cameras and monitoring devices). These devices are assumed to communicate with KGC on a secure channel. A KGC defines the preliminary setup as follows:

The KGC takes $k \in \mathbb{Z}_q^+$ as input and generates the public parameters and chooses its master secret key. The KGC selects an EC with a base point $G \in F_q$ over the finite field F_q according to the setup proposed in Section 3.5. The KGC generates its master secret key and the public parameters for the system as follows:

- Use the preliminary setup and determine the public parameters
- Choose the master key $x \in \mathbb{Z}_q^*$ uniformly at random and compute the system public key $Pu_{kgc} = x{\cdot}G$
- Choose the cryptographic hash functions $H_1 : \{0,1\}^* \times g_1 \times g_1 \to \mathbb{Z}_q, H_2 : g_1 \to \{0,1\}^{k_0}, H_3 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_4 : \{0,1\}^{k_0} \to \{0,1\}^k$, where k shows the fixed key length of a symmetric key and g_1 is a cyclic group generated by using the EC base point G.
- Keep the master key x as secret and publish the public parameters along with $Pu_{_{kgc}}$.

In the proposed system architecture, the smart cameras (senders) and the monitoring devices (receivers) share their public parameters with each other. This work only presents the key generation algorithms of smart cameras for the sake of simplicity. The same key generation procedure applies for the monitoring devices. The thesis provides a description of the used symbols in the glossary.

Request of Partial Private Key by a Smart Camera. Each smart camera j in the system architecture randomly chooses a secret value $Pr_j \in \mathbb{Z}_q^*$ and generates two public values using the base point G and the master public key Pu_{kgc} of KGC, e.g., $P_j = Pr_j{\cdot}G$ and $P_{j_{kgc}} = Pr_j{\cdot}Pu_{kgc}$. Then the smart camera sends the identity and public values $(j, P_j, P_{j_{kgc}})$ to the KGC to request a partial private key.

Partial Private Key Generation by KGC. Once the KGC receives the request from the smart camera for partial key generation, it first verifies its validity by checking if $P_{j_{kgc}} = x{\cdot}P_j$. If the validity is true then the KGC generates a partial private key, otherwise it rejects the request. The KGC generates the partial key by using its master secret key x, the identity j of the smart camera and the public parameters with permitted time period t_j as follows:

- Choose $r_j \in \mathbb{Z}_q^*$ randomly for each smart camera,
- Compute $X_j = r_j{\cdot}G$, $P_x = P_j + X_j$ and
- Compute $d_j = H_1(j, P_j, P_x)x + r_j \; mod \; q$.

The KGC sends the partial private key (d_j, X_j) to smart camera j via a secure channel.

Public Key of Smart Camera. As the smart camera receives its partial private key (d_j, X_j) from the KGC, it first verifies its validity and then generates the full public key as follows:

- Compute $P'_j = P_j + X_j$
- Compute $H_j = H_1(j, P_j, P'_j)$ and check if $d_j{\cdot}G = H_j Pu_{kgc} + X_j$ according to the Schnorr digital signature [55, 68]. Otherwise, reject the partial private key
- Compute $P''_j = H_j^{-1} P'_j$

The smart camera j chooses the full public key as $Pu_j = (j, P_j, P'_j, P''_j)$.

6.3 Operational Phase

In the operational phase, each smart camera initiates the signcryption process for the intended monitoring devices. The smart camera uses its private key Pr_j for the signature part, generates a session key for the encryption part and performs signcryption on the captured data as described in the following subsections.

Session Keys Generation. Each smart camera of a cluster participating in the surveillance of a specific area generates the symmetric keys for the encryption of data intended for the distinct monitoring devices as follows.

- The smart camera j chooses the public parameters and the list l of the identities of the monitoring devices.
- The smart camera j generates the symmetric key K_{enc} and the internal state information (e.g., firmware version and timestamp of the health status) represented by ω, using the public keys and other identity information of the monitoring devices.

The identity of smart camera j, the full public key Pu_j, the full private key Pr_j, the monitoring device identity h, the permitted time period t_h and the full public key Pu_h are given as inputs to the session key generation. The smart camera performs the following steps to get the symmetric key K_{enc} for all monitoring devices of a list l:

- The smart camera chooses $s_j \in \mathbb{Z}_q^*$ and $\sigma \in \{0,1\}^k$ randomly
- The smart camera then generates the session key as $K_{enc} = H_4(\sigma)$

Multi-Receiver Signcryption by Smart Cameras. Each smart camera signcrypts the region of interest ($RoI_{frames(f)}$) of frames f for the list l of monitoring devices with the relevant encryption key K_{enc} as follows:

- The region of interest is encrypted as $\varrho(l) = Enc_{K_{enc(l)}}(RoI_{frames(f)})$.
- The output of the required ciphertext for list l of monitoring devices is given as $\theta = (\varrho(1), \varrho(2), \varrho(3), \cdots \varrho(l))$.

Each smart camera uses the ciphertext data θ to complete the signcryption procedure with the following steps:

$$k_l = v \cdot G \tag{6.1}$$

$$r = H_3(\theta \| t_j \| k_l \| \omega_l \| l) \tag{6.2}$$

$$a_j = (H_j^{-1} d_j + r \cdot Pr_j + s_j) \; mod \; q \tag{6.3}$$

$$R = r \cdot G \tag{6.4}$$

$$H_h = H_1(h, P_h, P_h^{'}) \tag{6.5}$$

$$U_h = r H_h(Pu_{_{kgc}} + P_h^{''}) \tag{6.6}$$

$$V = \sigma \oplus H_2(R) \tag{6.7}$$

$$Signcryptext_j = (a_j, U_1, U_2 \ldots, U_h, V, \theta, l, k_l, \omega_l, t_j) \tag{6.8}$$

6.3.1 Multi-Receiver Aggregate-Signcryption Techniques

For multi-receiver aggregate-signcryption, the cluster head i performs the aggregation of all signcryptexts [155] from the smart cameras with other parameters for the sorting of relevant data. The sorting of $signcryptext_j$ is performed according to the identities h with relevant public keys Pu_h of the list l of monitoring devices. The cluster head verifies each signcryptexts with the public verification method as described in Section 6.3.2 and then uses the public keys of the smart cameras and monitoring devices to generate the aggregate-signcryptext as follows:

- Compute $S = \sum_{j=1}^{n} a_j$ and parse the θ according to the l, K_l, ω_l, t_j in a specific order.
- Merge the signcrypted data $(\theta(1) \cdots \theta(l), U_1 \cdots U_h, l, k_l, \omega_l, t_j, S)$ to the aggregate-signcryptext (ϑ) form.

6.3.2 Unsigncryption by Monitoring Devices

Unsigncryption is performed on the monitoring devices of the aggregate-signcryptexts after sorting them according to the list l. If the authentication and integrity of the data is verified, the decryption procedure is then applied to the given ciphertexts $\theta = (\varrho(l), \varrho(2), \varrho(3), \cdots \varrho(l))$ on each monitoring device. The decryption algorithm requires the full private and public keys of the monitoring devices with the public keys of the smart cameras to retrieve the decryption keys of the intended monitoring devices. The monitoring devices use the public keys of the smart camera $Pu_j = (j, P_j, P_j', P_j'')$ for the acceptance and verification of the signcrypted data.

- Find the corresponding U_h from the list l of the $signcryptext_j$.
- Compute $r^{'} = H_3(\theta \| t_j \| k_l \| \omega_l \| l)$.
- Compute $k_l^{'} = a_j \cdot G - ((r^{'} - H_j^{-1})P_j + P_j^{''} + Pu_{_{kgc}})$.
- Accept the signcrypted data, if $k_l^{'} = k_l$ and then verify the integrity of data ($U_h = r^{'} H_h(Pu_{_{kgc}} + P_h^{''})$).
- Proceed with the decryption if the data acceptance and verification was successful.
- Compute $R^{'} = (d_h + Pr_h)^{-1} U_h$.
- Compute $\sigma^{'} = V \oplus H_2(R^{'})$.
- Compute decryption key $K_{dec(l)} = H_4(\sigma^{'})$.
- Decrypt the ciphertext data to $RoI_{frames_f} = Dec_{K_{dec(l)}}(\theta)$ or apply decryption to the parsed ciphertexts e.g., $Dec_{K_{dec(l)}}(\theta(l))$= $Dec_{K_{dec(l)}}(\varrho(1), \varrho(2), \varrho(3), \cdots \varrho(l))$ data.

6.3.3 Correctness of the Scheme

As described in the unsigncryption process in Section 6.3.2, the $R^{'}$ value is needed to recover the relevant decryption keys on the monitoring devices for the decryption of the authentic ciphertexts. Only those monitoring devices can recover the decryption keys whose public information were already used in the aggregate-signcryption process, i.e., for the computation of R in Equ. (6.4). Therefore, now only those monitoring devices can use their associated private keys. The recovery of the correct decryption keys is based on the authentication and integrity of the aggregate-signcryptexts and on the associated private keys of the monitoring devices. These $R^{'}$ and R values should be equal to recover the correct decryption keys for the decryption of ciphertexts θ. The relevant monitoring devices prove the data integrity and authentication during unsigncryption by computing $k_l^{'}$ and then $r^{'}$. The $r^{'}$ value is further used in the computation of U_h which provides $R^{'}$ (e.g., $R^{'} = (d_h + Pr_h)^{-1}U_h$). Therefore, the proof of $R^{'} = R$ can be described as follows:

$$\begin{aligned}
R^{'} &= (d_h + Pr_h)^{-1}U_h \\
&= \frac{U_h}{(d_h + Pr_h)} \\
&= \frac{rH_h(Pu_{_{kgc}} + P_h^{''})}{(H_1(h, P_h, P_h^{'})x + x_h) + Pr_h} \\
&= \frac{r(H_h(Pu_{_{kgc}}) + H_h{\cdot}H_h^{-1}P_h^{'})}{(H_1(h, P_h, P_h^{'})x + x_h) + Pr_h} \\
&= \frac{r(H_h(Pu_{_{kgc}}) + (Pu_h + P_h))}{(H_1(h, P_h, P_h^{'})x + x_h) + Pr_h} \\
&= \frac{r(H_1(h, P_h, P_h^{'})x{\cdot}G + (Pr_h + x_h){\cdot}G)}{H_1(h, P_h, P_h^{'})x + x_h + Pr_h} \\
&= \frac{r(H_1(h, P_h, P_h^{'})x + Pr_h + x_h){\cdot}G}{H_1(h, P_h, P_h^{'})x + Pr_h + x_h} \\
&= r{\cdot}G
\end{aligned}$$

The $R^{'}$ value is further used in the computation of $\sigma^{'}$ that should be equal to the value of σ (which was already computed for the generation of encryption keys on smart cameras side). The correctness can thus be shown as follows:

$$\begin{aligned}
\sigma^{'} &= V \oplus H_2(R^{'}) \\
&= \sigma \oplus H_2(R) \oplus H_2(R^{'}) \\
&= \sigma \oplus H_2(r{\cdot}G) \oplus H_2(r{\cdot}G) \\
&= \sigma
\end{aligned}$$

Hence, the list of decryption keys can be recovered on each monitoring device according to their own information as follows:

$$\begin{aligned} K_{dec(l)} &= H_4(\sigma^{'}) \\ &= H_4(\sigma) \\ &= K_{enc(l)} \end{aligned}$$

Data authentication and replay attack prevention can be checked as follows: First, calculate the value of $a_j{\cdot}G$.

$$\begin{aligned} hj &= H_1(j, P_j, P_j^{'}) \\ a_j{\cdot}G &= H_j^{-1}d_j{\cdot}G + Pu_{_{kgc}}{\cdot}G + v{\cdot}G \\ &= H_j^{-1}(H_j s{\cdot}G + x_j{\cdot}G) + r{\cdot}P_j + v{\cdot}G \\ &= x{\cdot}G + H_j^{-1}x_j{\cdot}G + r{\cdot}P_j + v{\cdot}G \\ &= Pu_{_{kgc}} + H_j^{-1}x_j{\cdot}G + r{\cdot}P_j + v{\cdot}G \end{aligned}$$

Second, calculate the value of $(r^{'} - H_j^{-1})P_j + P_j^{''} + Pu_{_{kgc}}$:

$$\begin{aligned} (r^{'} - H_j^{-1})P_j + P_j^{''} + Pu_{_{kgc}} &= (r^{'} - H_j^{-1})P_j + H_j^{-1}(P_j + x_j{\cdot}G) + Pu_{_{kgc}} \\ &= r^{'}{\cdot}P_j + H_j^{-1}x_j{\cdot}G + Pu_{_{kgc}} \end{aligned}$$

Finally, the value of $(r^{'} - H_j^{-1})P_j + P_j^{''} + Pu_{_{kgc}}$ must be subtracted from $a_j{\cdot}G$ resulting in k_l, which shows the authenticity and proof of the prevention of the replay attack:

$$\begin{aligned} a_j{\cdot}G - [(r^{'} - H_j^{-1})P_j + P_j^{''} + Pu_{_{kgc}}] &= v{\cdot}G \\ &= k_l \end{aligned}$$

6.4 Security Analysis

The multi-receiver aggregate-signcryption scheme for the proposed system architecture (Figure 3.5) provides the basic security properties, e.g., public verifiability, authentication, integrity, freshness, confidentiality, decryption fairness and forward secrecy for the captured data received from smart cameras. These properties are briefly analyzed in the following sections.

Public Verification. The security technique of multi-receiver aggregate-signcryption provides the public verifiability of the data by any trusted or untrusted entity in the system without decryption of the data. The public verifiability proves, if $U_j = r^{'} H_j(Pu_{_{kgc}} + P_j^{''})$ is true for smart camera j. The advantage of the public verifiability is that the authenticity of the data received from the source can be proven at any stage by a trusted or untrusted entity. The public verification does not require the private keys of smart cameras, the verification is possible with the relevant public information of the devices.

Authentication and Integrity. The authentication can be checked by the intended receiver by computing $k_l^{'} = a_j{\cdot}G{-}((r^{'}{-}H_j^{-1})P_j{+}P_j^{''}{+}Pu_{_{kgc}})$ and then comparing it with the received value of k_l from the smart camera. $k_l^{'} = k_l$ means that the data comes from an authentic smart camera. The integrity of the received data can also be verified by $k_l^{'} = k_1$ because in the computation of k_l the value of $r = H_3(\theta \| t \| k_l \| \omega_l \| l)$ is required, which is the hashed valued of the ciphertext data θ along with the other information. The r value is further multiplied with the secret key of the smart camera (Pr_j), as its public key is used for the verification purpose in the computation of $k_l^{'}$. Therefore, the proof for $k_l^{'} = k_l$ provides both the properties of integrity and authenticity of the received data. The attacker cannot compromise the integrity and authenticity without the private key of smart camera Pr_j and guessing of a private key from the public key is a hard problem because of the ECDLP assumption.

Decryption Fairness and Confidentiality. Decryption fairness is the capability of the monitoring device to extract the decryption key from shared information using their own credentials. Confidentiality of data is the prevention of access from unauthorized users and the guarantee of exclusive access for the intended receivers. Only the intended receivers can exclusively access the shared information for their own decryption key recovery with the help of their private keys. None of the monitoring devices other than intended can recover the decryption key to access the data for another monitoring device because the private key associated with the public keys is hard to guess due to the assumption of ECDLP.

Freshness. The smart camera uses the hash of the timestamp in the computation of r along with the concatenated value of the ciphertext. This guarantees the freshness of the ciphertext data. If the timestamp value is compromised by an attacker, then the computation of $r^{'}$ on monitoring device results in an incorrect value because of the collision resistance of the hash function, and the authenticity and integrity proof fails.

6.5 Experimental Evaluation

In this section, this work presents the experimental setup and investigate the computational effort. In the deployment phase, we measure the computation time of key generation and verification. In the operational phase, this work measures and compares the computation times and communication overheads for the individual-signcryption, aggregate-signcryption and multi-receiver aggregate-signcryption approaches.

6.5.1 Experimental Setup

This thesis has prototypically implemented the multi-sender/multi-receiver scenario of the system architecture (Figure 3.5), where standard laptops (Intel core i5 with 2.6 GHz and 8 GB RAM) running Windows 10 serve as a platform for the key devices (e.g., smart cameras, cluster head, monitoring devices and KGC). We used a standard laptop for ease of implementation and fair comparisons of the different approaches. Runtime measurements for the individual- and aggregate-signcryption have been performed on embedded platforms in our previous work (cp. Chapters 4 and 5).

In these experiments, each camera signcrypts 25 images, where each image has a predefined QVGA resolution (320 × 240 pixels) and size of 30 kB. The open source library BouncyCastle [3] is used for the implementation of the EC-based signcryption algorithm and used the EC-finite field of P-384 and a 256 bit AES key. The proposed techniques are implemented in Java due to readily available libraries for the main cryptographic building blocks. We run the application with the same configuration ten times for key generation, signcryption, aggregate-signcryption and multi-receiver aggregate-signcryption for each device and recorded their average running time.

6.5.2 Deployment Phase

In the deployment phase, the KGC initiates the system setup and shares the public parameters among all participating devices (i.e., the smart cameras and monitoring devices). Each device uses those public parameters, chooses a private key, and sends a request for partial key generation to the KGC. The KGC verifies the request and generates a partial key for the requesting device. After the KGC has sent the partial key to the requesting device, it first verifies its authenticity. The requesting device further generates a full public key based on their partial private key and public parameters. The computation times of these steps are summarized in Table 6.1. The KGC only requires 20.2 ms for its

Table 6.1: Keys generation and verification time (in ms) in the deployment phase. Legend: SC: smart camera, MD: monitoring device, KGC: key generation center, Pa: partial, Pu: public, TT: total time. The symbol – indicates that the required action is not performed on the corresponding device for the key generation or verification.

	Computational Time (all in [ms])					
Devices	Generation Algorithm			Verification Algorithm		TT
	Pa-key-request	Pu-key	Pa-key	Pa-key	Pa-key-request	
SC	100.7	85.2	–	31.9	–	217.8
MD	100.3	84.7	–	32.2	–	217.2
KGC	–	20.2	47	–	32.3	99.5

public key generation while the smart cameras and monitoring devices require more time because of their full public key generation based on their partial and private keys. The total computation time required to generate a full public key with the help of the KGC using a certificateless approach is 217.8 ms on a smart camera, 217.2 ms on a monitoring device and 99.5 ms on the KGC platform.

6.5.3 Operational Phase

In the operational phase, the smart cameras monitor a specific area, capture relevant information in the form of images and perform signcryption to secure it for a single device or multiple monitoring devices. We evaluate the computation time and communication overheads for individual-, aggregate- and multi-receiver aggregate-signcryption with different numbers of senders (m smart cameras) and receivers (n monitoring devices). We measure the computation times of the individual steps of each approach and compare the total runtimes based on three scenarios: single-sender/single-receiver (1-1), multi-sender/single-receiver (m-1) and multi-sender/multi-receiver (m-n).

Computational Time of Individual-Signcryption. Table 6.2 depicts the measured computational times for the key steps of individual-signcryption: signcryption, verification and decryption. These times have been measured for five different cases (C1 to C5). Here each camera individually signcrypts the captured images and transfers them to the monitoring device where each signcrypted data is verified and decrypted sequentially. In the case of multiple senders, the cameras operate in parallel, thus the maximum signcryption time limits the total time on the sender. On the receiver, the received signcrypted data must be processed sequentially. In the case of multiple receivers, each camera must separately signcrypt the images for each receiver. The total time for individual-signcryption can be estimated as follows

$$TT = n \cdot (max_m(ST)) + max_n(m \cdot (VT + DT)) \tag{6.9}$$

where max_m and max_n represent the longest time among m smart cameras and n monitoring devices, respectively.

Table 6.2: Computational time for individual-signcryption. Legend: SC: smart camera, MD: monitoring device, ST: signcryption time, VT: verification time, DT: decryption time.

Id	SC	MD	
	ST [ms]	VT [ms]	DT [ms]
C1	320.5	155.4	283.8
C2	321.0	154.7	285.0
C3	319.8	156.0	284.7
C4	321.3	155.3	286.2
C5	320.7	154.8	283.9

Computational Time of Aggregate-Signcryption. Table 6.3 depicts the measured computational times for the key steps of aggregate-signcryption: signcryption, aggregation, verification and decryption. These times have been measured in a cluster of five cameras that send their signcrypted images to the cluster head for aggregation. The cluster head then transfers the aggregated data to the monitoring device where only a single verification is necessary. In the case of multiple receivers, still each camera separately signcrypts the images for each receiver. Thus, the total time for aggregate signcryption can be estimated as follows

$$TT = n \cdot (max_m(ST) + AT) + max_n(VT + m \cdot DT). \tag{6.10}$$

Table 6.3: Computational time for aggregate-signcryption. Legend: SC: smart camera, CH: cluster head, MD: monitoring device, ST: signcryption time, AT: aggregation time, VT: verification time, DT: decryption time.

Id	SC	CH	MD	
	ST [ms]	AT [ms]	VT [ms]	DT [ms]
C1	320.7	145.5	160.3	284.9
C2	321.2			285.0
C3	319.9			286.3
C4	321.1			284.9
C5	322.0			285.4

Computational Time of Multi-Receiver Aggregate-Signcryption. Table 6.4 depicts the measured computational times for the key steps of multi-receiver aggregate-signcryption: signcryption, aggregation, verification and decryption. These times have also been measured in a cluster of five cameras. Please note that in this approach, signcryption and aggregation are more complex than in the other approaches but no separate signcryption is required for each receiver in the case of multiple monitoring devices. Thus, the total time can be estimated as follows

$$TT = max_m(ST) + AT + max_n(VT + m \cdot DT). \tag{6.11}$$

Table 6.4: Computational time for multi-receiver aggregate-signcryption. Legend: SC: smart camera, CH: cluster head, MD: monitoring device, ST: signcryption time, AT: aggregation time, VT: verification time, DT: decryption time.

Id	SC	CH	MD	
	ST [ms]	AT [ms]	VT [ms]	DT [ms]
C1	345.4	166.2	172.4	288.3
C2	346.0			287.7
C3	345.2			286.9
C4	344.9			288.5
C5	345.5			287.6

Performance Comparison. Table 6.5 compares the total times of our three approaches based on three scenarios: one sender and one receiver (1-1), five senders and one receiver (5-1) and five senders and three receivers (5-3). The total times are based on the measured run times of the corresponding approaches and Equs. (6.9)–(6.11). We highlighted the most efficient approach for each scenario in gray.

Table 6.5: Comparisons of total times (in ms) of different approaches for one smart camera/one monitoring device (1-1), five smart cameras/one monitoring device (5-1) and five smart cameras and three monitoring devices (5-3). Legend: IS: individual-signcryption, AS: aggregate-signcryption, MAS: multi-receiver aggregate-signcryption.

Scenario	IS	AS	MAS
1-1	759.7	911.4	972.3
5-1	2521.1	2054.3	2123.6
5-3	3166.6	2989.0	2124.9

As expected, individual-signcryption is most efficient for the 1-1 scenario due to the low overhead. Aggregate-signcryption is superior for the 5-1 scenario, since it avoids multiple verifications on the receiver. Finally, multi-receiver aggregate-signcryption is the best option for scenario 5-3, where it shows a reduction of 32.89% and 28.90% as compared to individual-signcryption and aggregate-signcryption, respectively.

Communication Efficiency. Table 6.6 presents the communication efficiency of the three approaches of this work by comparing the total amount of transferred data and the number of necessary data transfers. In the experimental setup of this work, each smart camera signcrypts 25 images which a total size of 750 kB. This work uses the AES 256 bit encryption scheme in CBC mode which results in a ciphertext of same size as the input data.

In the 1-1 scenario, the individual-, aggregate- and multi-receiver aggregate-signcryption transfer 750 kB of ciphertext data. The individual- and aggregate-signcryption require extra data of 72 Bytes for the signature part, while multi-receiver aggregate-signcryption requires extra data of 340 Bytes because additional parameters are needed for the verification of the signature to enable the multi-receiver setup. Aggregate-signcryption and multi-receiver aggregate-signcryption require an additional transfer to the cluster head.

In the 5-1 scenario, the five smart cameras send their protected data to a single monitoring device, so the individual-, aggregate-, and multi-receiver aggregate-signcryption send the same amount of ciphertext data (e.g., 3750 kB), while the size of the extra data varies for each case. Individual-signcryption requires 360 Bytes for the individual verification of the signcryptexts. Aggregate-signcryption performs only a single verification and reduces the extra data to 168 Bytes. Multi-receiver aggregate-signcryption requires 340 Byte to enable the decryption of same data for multiple monitoring devices (which are actually not required in single-receiver scenarios). Aggregate- and multi-receiver aggregate-signcryption require five transfers to the cluster head and one transfer to the monitoring device or backup server.

In the 5-3 scenario, the individual- and aggregate-signcryption protect the same data three times for the three different monitoring devices which aggregates to 11,250 kB of ciphertext data. Similarly, extra data must be separately included for each receiver. However, multi-receiver aggregate-signcryption sends the same ciphertext to all monitoring devices and needs only 24 Bytes for each monitoring device in addition to the single-receiver extra data. Similar to the computational efficiency, individual-signcryption is most communication efficient for the 1-1 scenario due to the low overhead. Aggregate-signcryption is superior for the

5-1 scenario, since it avoids multiple signatures and verifications. Finally, multi-receiver aggregate-signcryption is the best option for scenario 5-3.

Table 6.6: Comparisons of communication efficiency in terms of transferred data and number of data transfers of different approaches for one smart camera/one monitoring device (1-1), five smart cameras/one monitoring device (5-1) and five smart cameras/three monitoring devices (5-3). Legend: IS: individual-signcryption, AS: aggregate-signcryption, MAS: multi-receiver aggregate-signcryption, SD: signcryptext data, CD: ciphertext data, ED: extra data for signature and verification, NT: number of transfers.

Scenario	IS			AS			MAS		
	SD		NT	SD		NT	SD		NT
	CD [kB]	ED [B]		CD [kB]	ED [B]		CD [kB]	ED [B]	
1-1	750	72	1	750	72	2	750	340	2
5-1	3750	360	5	3750	168	6	3750	340	6
5-3	11250	1080	15	11250	504	18	3750	388	8

Chapter 7

Conclusion and Future Work

This chapter concludes the thesis by a summary of its contributions and an outlook to future research directions.

7.1 Conclusion

In this thesis, the EC-based signcryption has been implemented and protected the data captured of smart cameras for event-triggered monitoring in IoT applications. This work 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. The evaluation of aggregate-signcryption is performed and implemented the EC-based signcryption for the security of multiple images on a smart camera and reduced the average running time per image. Second, this thesis investigated the performance of aggregate-signcryption for cluster-based smart camera IoT applications. This work implemented the aggregate-signcryption and investigated the performance in cluster-based multi-camera network and reduced the communication and computation overheads. This work also evaluated the performance ratio between individual and aggregate-signcryption in term of communication and computation overhead in multi-sender/single-receiver communication scenario. Third, this thesis investigated certificateless key generation technique and lightweight multi-receiver aggregate signcryption for cluster-based smart camera IoT applications. This work adopted EC-based signcryption for each smart camera to achieve end-to-end lifetime data security. This work implemented aggregation on cluster heads to merge the signcryptexts as a multi-receiver aggregate-signcryptext packet and to avoid the transfer of unnecessary extra data. This work performed unsigncryption on each monitoring device with public verifiability and exclusive access to the encrypted data. Finally, in the experimental evaluation, this work explored the computation and communication effort of individual-, aggregate- and multi-receiver aggregate-signcryption on three different sender/receiver scenarios. The resource consumption for these three approaches is investigated for single-sender/single-receiver, multi-sender/single-receiver and multi-sender/multi-receiver scenarios.

7.2 Future Work

The 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. The current work initiates the data transfer when simple pre-defined events have been detected and the 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. We further foresee several directions for future work including investigating alternative encryption approaches for signcryption, its implementation on embedded smart camera platforms and its deployments in a smart home case study.

Bibliography

[1] https://trusteye.nes.aau.at/. [Last accessed: 27-03-2019].

[2] https://www.raspberrypi.org/. [Last accessed: 27-03-2019].

[3] https://www.bouncycastle.org/. [Last accessed: 27-03-2019].

[4] K. Abas, C. Porto, and K. Obraczka. Wireless smart camera networks for the surveillance of public spaces. *Computer*, 47(5):37–44, May 2014.

[5] Kevin Abas, Katia Obraczka, and Leland Miller. Solar-powered, wireless smart camera network: An iot solution for outdoor video monitoring. *Computer Communications*, 118:217 – 233, March 2018.

[6] Aditya, M. Sharma, and S. Chand Gupta. An internet of things based smart surveillance and monitoring system using arduino. In *Proc. International Conference on Advances in Computing and Communication Engineering (ICACCE)*, pages 428–433, June 2018.

[7] Hamid Aghajan and Andrea Cavallaro. *Multi-Camera Networks: Principles and Applications.* Academic Press, 2009.

[8] Mayssaa Al Najjar, Milad Ghantous, and Magdy Bayoumi. *Visual Sensor Nodes In: Video Surveillance for Sensor Platforms. Lecture Notes in Electrical Engineering, vol 114.*, pages 17–35. Springer, New York, 2014.

[9] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless public key cryptography. In Chi-Sung Laih, editor, *Advances in Cryptology - ASIACRYPT*, pages 452–473, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[10] M. Alam, J. Ferreira, S. Mumtaz, M. A. Jan, R. Rebelo, and J. A. Fonseca. Smart cameras are making our beaches safer: A 5g-envisioned distributed architecture for safe, connected coastal areas. *IEEE Vehicular Technology Magazine*, 12(4):50–59, Dec 2017.

[11] S. Alam, A. Jamil, A. Saldhi, and M. Ahmad. Digital image authentication and encryption using digital signature. In *Proc. International Conference on Advances in Computer Engineering and Applications*, pages 332–336, March 2015.

[12] Pietro Albano, Andrea Bruno, Bruno Carpentieri, Aniello Castiglione, Arcangelo Castiglione, Francesco Palmieri, Raffaele Pizzolante, and Ilsun You. A secure distributed video surveillance system based on portable devices. In Gerald Quirchmayr, Josef Basl, Ilsun You, Lida Xu, and Edgar Weippl, editors, *Multidisciplinary Research and Practice for Information Systems*, pages 403–415, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[13] Rana Alharbi and David Aspinall. An iot analysis framework: An investigation of iot smart cameras' vulnerabilities. In *Proc. Living in the Internet of Things: Cybersecurity of the IoT*, pages 1–10. IET, 2018.

[14] I. Alqassem and D. Svetinovic. A taxonomy of security and privacy requirements for the internet of things (iot). In *Proc. IEEE International Conference on Industrial Engineering and Engineering Management*, pages 1244–1248, Dec 2014.

[15] Ali Alqazzaz, Ibrahim Alrashdi, Esam Aloufi, Mohamed Zohdy, and Hua Ming. SecSPS: A Secure and Privacy-Preserving Framework for Smart Parking Systems. *Journal of Information Security*, 09(04):299–314, 2018.

[16] Mohammed Alshahrani and Issa Traore. Secure mutual authentication and automated access control for iot smart home using cumulative keyed-hash chain. *Journal of Information Security and Applications*, 45:156 – 175, 2019.

[17] Mohammad A. Alsmirat, Yaser Jararweh, Islam Obaidat, and Brij B. Gupta. Internet of surveillance: a cloud supported large-scale wireless surveillance system. *The Journal of Supercomputing*, 73(3):973–992, Mar 2017.

[18] Mohammad A. Alsmirat, Islam Obaidat, Yaser Jararweh, and Mohammed Al-Saleh. A security framework for cloud-based video surveillance system. *Multimedia Tools and Applications*, 76(21):22787–22802, Nov 2017.

[19] A. A. Altahir, V. S. Asirvadam, N. H. Hamid, P. Sebastian, N. Saad, R. Ibrahim, and S. C. Dass. Modeling multicamera coverage for placement optimization. *IEEE Sensors Letters*, 1(6):1–4, Dec 2017.

[20] A. A. Altahir, V. S. Asirvadam, N. H. B. Hamid, P. Sebastian, N. B. Saad, R. B. Ibrahim, and S. C. Dass. Optimizing visual surveillance sensor coverage using dynamic programming. *IEEE Sensors Journal*, 17(11):3398–3405, June 2017.

[21] M.S. Anoop. Elliptic Curve Cryptography. *Infosecwriters*, pages 1–11, 2015.

[22] Noah Apthorpe, Dillon Reisman, and Nick Feamster. A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic. *arXiv preprint arXiv:1705.06805*, 2017.

[23] Pradeep K. Atrey, Wei-Qi Yan, and Mohan S. Kankanhalli. A scalable signature scheme for video authentication. *Multimedia Tools and Applications*, 34(1):107–135, 2007.

[24] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.

[25] Remigiusz Baran, Tomasz Rusc, and PawełFornalski. A smart camera for the surveillance of vehicles in intelligent transportation systems. *Multimedia Tools Appl.*, 75(17):10471–10493, September 2016.

[26] M. Barbosa and P. Farshim. Certificateless signcryption. In *Proc. of the ACM Symposium on Information, Computer and Communications Security*, ASIACCS, pages 369–372, New York, NY, USA, 2008. ACM.

[27] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology — CRYPTO '96*, pages 1–15, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[28] Merwan Birem and Franois Berry. Dreamcam: A modular fpga-based smart camera architecture. *Journal of Systems Architecture*, 60(6):519 – 527, 2014.

[29] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Proc. Advances in Cryptology — CRYPTO*, pages 213–229, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[30] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *Journal of Cryptology*, 17(4):297–319, Sep 2004.

[31] Mokrane Bouzeghoub. A framework for analysis of data freshness. In *Proc. of the 2004 International Workshop on Information Quality in Information Systems*, IQIS '04, pages 59–67, New York, NY, USA, 2004. ACM.

[32] An Braeken, Pawani Porambage, Andrei Gurtov, and Mika Ylianttila. Proc. secure and efficient reactive video surveillance for patient monitoring. *Sensors*, 16(1), 2016.

[33] M. Brezovan and C. Badica. A review on vision surveillance techniques in smart home environments. In *Proc. International Conference on Control Systems and Computer Science*, pages 471–478, May 2013.

[34] 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*, 62(11):2629–2640, Nov 2015.

[35] 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*, 62(11):2629–2640, Nov 2015.

[36] L. Catarinucci, D. de Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone. An iot-aware architecture for smart healthcare systems. *IEEE Internet of Things Journal*, 2(6):515–526, Dec 2015.

[37] S. Chien, W. Chan, Y. Tseng, C. Lee, V. Somayazulu, and Y. Chen. Distributed computing in iot, system-on-a-chip for smart cameras as an example. In *Proc. 20th Asia and South Pacific Design Automation Conference*, pages 130–135, Jan 2015.

[38] Wayne Chiu, Chunhua Su, Chuan-Yen Fan, Chien-Ming Chen, and Kuo-Hui Yeh. Authentication with what you see and remember in the internet of things. *Symmetry*, 10(11), 2018.

[39] C. S. Collberg and C. Thomborson. Watermarking, tamper-proofing, and obfuscation - tools for software protection. *IEEE Transactions on Software Engineering*, 28(8):735–746, Aug 2002.

[40] A. Costache, D. Popescu, C. Popa, and S. Mocanu. Efficient video monitoring of areas of interest. In *Proc. 26th Telecommunications Forum (TELFOR)*, pages 1–4, Nov 2018.

[41] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[42] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES-The Advanced Encryption Standard*. Springer Science & Business Media, 2002.

[43] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013.

[44] Oscar Deniz, Noelia Vallez, Jose L. Espinosa-Aranda, Jose M. Rico-Saavedra, Javier Parra-Patino, Gloria Bueno, David Moloney, Alireza Dehghani, Aubrey Dunne, Alain Pagani, Stephan Krauss, Ruben Reiser, Martin Waeny, Matteo Sorci, Tim Llewellynn, Christian Fedorczak, Thierry Larmoire, Andre Herbst, Marcoand Seirafi, and Kasra Seirafi. Eyes of things. *Sensors*, 17(5), 2017.

[45] Alexander W. Dent. A survey of certificateless encryption schemes and security models. *International Journal of Information Security*, 7(5):349–377, Oct 2008.

[46] S. Dietzel, A. Peter, and F. Kargl. Secure cluster-based in-network information aggregation for vehicular networks. In *Proc. IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–5, May 2015.

[47] A. Dillon. https://github.com/Hopding/JRPiCam. [Last accessed: 27-03-2019].

[48] A. Duluta, S. Mocanu, R. Pietraru, D. Merezeanu, and D. Saru. Secure communication method based on encryption and steganography. In *Proc. 21st International Conference on Control Systems and Computer Science (CSCS)*, pages 453–458, May 2017.

[49] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, July 1985.

[50] Adam Erdelyi, Tibor Barat, Patrick Valet, Thomas Winkler, and Bernhard Rinner. Adaptive Cartooning for Privacy Protection in Camera Networks. In *Proc. IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, pages 44–49, 2014.

[51] Junfeng Fan, Kazuo Sakiyama, and Ingrid Verbauwhede. Elliptic curve cryptography on embedded multicore systems. *Design Automation for Embedded Systems*, 12(3):231–242, 2008.

[52] B. Feng, W. Lu, and W. Sun. Secure binary image steganography based on minimizing the distortion on the texture. *IEEE Transactions on Information Forensics and Security*, 10(2):243–255, Feb 2015.

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

[54] S. Fleck and W. Strasser. Smart camera based monitoring system and its application to assisted living. *Proc. of the IEEE*, 96(10):1698–1714, Oct 2008.

[55] David Mandell Freeman. Schnorr Identification and Signatures. *October 20*, pages 2–5, 2011.

[56] Yanfeng Geng and Christos G Cassandras. New smart parking system based on resource allocation and reservations. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1129–1139, 2013.

[57] R. Goshorn, J. Goshorn, D. Goshorn, and H. Aghajan. Architecture for cluster-based automated surveillance network for detecting and tracking multiple persons. In *Proc. of First ACM/IEEE International Conference on Distributed Smart Cameras*, pages 219–226, Sept 2007.

[58] S. Greene, H. Thapliyal, and D. Carpenter. Iot-based fall detection for smart home environments. In *Proc. IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*, pages 23–28, Dec 2016.

[59] K. Grgic, V. Mendelski, and D. Zagar. Security framework for visual sensors and smart camera networks. In *Proc. 14th International Conference on Telecommunications (ConTEL)*, pages 131–138, June 2017.

[60] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645 – 1660, 2013.

[61] I. Haider and B. Rinner. Private space monitoring with soc-based smart cameras. In *Proc. IEEE 14th International Conference on Mobile AdHoc and Sensor Systems (MASS)*, pages 19–27, Oct 2017.

[62] Darrel Hankerson, Alfred J Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography.* Springer Science & Business Media, 2006.

[63] L. Harn and Y. Xu. Design of generalised elgamal type digital signature schemes based on discrete logarithm. *Electronics Letters*, 30(24):2025–2026, Nov 1994.

[64] Latifah Uswatun Hasanah, Tito Waluyo Purboyo, and Randy Erfa Saputra. A review of mp3 steganography methods. *International Journal of Applied Engineering Research*, 13(2):1128–1133, 2018.

[65] Stephan Hengstler, Daniel Prashanth, Sufen Fong, and Hamid Aghajan. Mesheye: A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In *Proc. of the 6th International Conference on Information Processing in Sensor Networks*, IPSN '07, pages 360–369, New York, NY, USA, 2007. ACM.

[66] Naoki Hosoe, Kaoru Takabayashi, Haruhiko Ogata, and Takanori Kanai. Capsule endoscopy for small-intestinal disorders: current status. *Digestive Endoscopy*, Jan 2019.

[67] C. Huang, R. Lu, X. Lin, and X. Shen. Secure automated valet parking: A privacy-preserving reservation scheme for autonomous vehicles. *IEEE Transactions on Vehicular Technology*, 67(11):11169–11180, Nov 2018.

[68] Alin Ionut. Elliptic curves differentiation with application to group signature scheme. *Electronic Journal of Differential Equations*, 2017(237):1–21, 2017.

[69] Hao Jin, Ke Zhou, Hong Jiang, Dongliang Lei, Ronglei Wei, and Chunhua Li. Full integrity and freshness for cloud data. *Future Generation Computer Systems*, 80:640 – 652, 2018.

[70] Zhengping Jin, Qiaoyan Wen, and Hua Zhang. A supplement to liu et al.'s certificateless signcryption scheme in the standard model. *IACR cryptology ePrint Archive*, 2010:252, 2010.

[71] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International Journal of Information Security*, 1(1):36–63, Aug 2001.

[72] A. Jurisic and A. Menezes. Elliptic curves and cryptography. *Dr. Dobbs Journal*, pages 26–36, 1997.

[73] J. Karsek, R. Burget, and O. Morsk. Towards an automatic design of non-cryptographic hash function. In *Proc. 34th International Conference on Telecommunications and Signal Processing (TSP)*, pages 19–23, Aug 2011.

[74] Aliaksei Kerhet, Michele Magno, Francesco Leonardi, Andrea Boni, and Luca Benini. A low-power wireless video sensor node for distributed object detection. *Journal of Real-Time Image Processing*, 2(4):331–342, Dec 2007.

[75] Q. Kester, L. Nana, and A. C. Pascu. A novel cryptographic encryption technique for securing digital images in the cloud using aes and rgb pixel

displacement. In *Proc. European Modelling Symposium*, pages 293–298, Nov 2013.

[76] M. Khabbazian, T. A. Gulliver, and V. K. Bhargava. Double point compression with applications to speeding up random point multiplication. *IEEE Transactions on Computers*, 56(3):305–313, March 2007.

[77] A. Khan, B. Rinner, and A. Cavallaro. Cooperative robots to observe moving targets: Review. *IEEE Transactions on Cybernetics*, 48(1):187–198, Jan 2018.

[78] George Kokkonis, Kostas E. Psannis, Manos Roumeliotis, and Dan Schonfeld. Real-time wireless multisensory smart surveillance with 3d-hevc streams for internet-of-things (iot). *The Journal of Supercomputing*, 73(3):1044–1062, Mar 2017.

[79] Jia Hao Kong, Li-Minn Ang, and Kah Phooi Seng. A comprehensive survey of modern symmetric cryptographic solutions for resource constrained environments. *Journal of Network and Computer Applications*, 49(Supplement C):15 – 50, 2015.

[80] R Kroijer, M Kobaek-Larsen, N Qvist, T Knudsen, and G Baatrup. Colon capsule endoscopy for colonic surveillance. *Colorectal Disease*, Jan.

[81] C. Lee, J. Shen, and Z. Chen. A survey of watermarking-based authentication for digital image. In *Proc. 3rd International Conference on Computer and Communication Systems (ICCCS)*, pages 207–211, April 2018.

[82] Donghyeok Lee and Namje Park. Geocasting-based synchronization of almanac on the maritime cloud for distributed smart surveillance. *The Journal of Supercomputing*, 73(3):1103–1118, Mar 2017.

[83] Arjen K Lenstra and Eric R Verheul. Selecting cryptographic key sizes. *Journal of cryptology*, 14(4):255–293, 2001.

[84] Fagen Li, Yanan Han, and Chunhua Jin. Practical signcryption for secure communication of wireless sensor networks. *Wireless Personal Communications*, 89(4):1391–1412, 2016.

[85] J. Lim, J. Seo, and Y. Baek. Camthings: Iot camera with energy efficient communication by edge computing based on deep learning. In *Proc. 28th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–6, Nov 2018.

[86] B. Liu, X. L. Yu, S. Chen, X. Xu, and L. Zhu. Blockchain based data integrity service framework for iot data. In *Proc. IEEE International Conference on Web Services (ICWS)*, pages 468–475, June 2017.

[87] He Liu, Stefan Saroiu, Alec Wolman, and Himanshu Raj. Software abstractions for trusted sensors. In *Proc. of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 365–378, New York, NY, USA, 2012. ACM.

[88] Hong Liu, Huansheng Ning, Qitao Mu, Yumei Zheng, Jing Zeng, Laurence T. Yang, Runhe Huang, and Jianhua Ma. A review of the smart world. *Future Generation Computer Systems*, 2017.

[89] Junbin Liu, Sridha Sridharan, and Clinton Fookes. Recent advances in camera planning for large area surveillance: A comprehensive review. *ACM Comput. Surv.*, 49(1):6:1–6:37, May 2016.

[90] T. Ma, M. Hempel, D. Peng, and H. Sharif. A survey of energy-efficient compression and communication techniques for multimedia in resource constrained systems. *IEEE Communications Surveys and Tutorials*, 15(3):963–972, March 2013.

[91] Aaron Mavrinac and Xiang Chen. Modeling coverage in camera networks: A survey. *International Journal of Computer Vision*, 101(1):205–226, Jan 2013.

[92] Kevin S McCurley. The discrete logarithm problem. In *AMS Proc. Symp. Appl. Math*, volume 42, pages 49–74, 1990.

[93] M. Mehrubeoglu and R. Muddu and. Real-time eye tracking using a smart camera. In *Proc. IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–7, Oct 2011.

[94] L. Meinel, M. Findeisen, M. He, A. Apitzsch, and G. Hirtz. Automated real-time surveillance for ambient assisted living using an omnidirectional camera. In *Proc. IEEE International Conference on Consumer Electronics (ICCE)*, pages 396–399, Jan 2014.

[95] Elsayed Mohamed and Hassan Elkamchouchi. Elliptic curve signcryption with encrypted message authentication and forward secrecy. *International Journal of Computer Science and Network Security*, 9(1):395–398, 2009.

[96] S. P. Mohanty. A secure digital camera architecture for integrated real-time digital rights management. *Journal of Systems Architecture*, 55(10–12):468–480, 2009.

[97] S. P. Mohanty, U. Choppali, and E. Kougianos. Everything you wanted to know about smart cities: The internet of things is the backbone. *IEEE Consumer Electronics Magazine*, 5(3):60–70, July 2016.

[98] Bassam J. Mohd, Thaier Hayajneh, and Athanasios V. Vasilakos. A survey on lightweight block ciphers for low-resource devices: Comparative study and open issues. *Journal of Network and Computer Applications*, 58(Supplement C):73–93, 2015.

[99] Peter L Montgomery. A survey of modern integer factorization algorithms. *CWI quarterly*, 7(4):337–366, 1994.

[100] Higinio Mora, David Gil, Rafael Muoz Terol, Jorge Azorn, and Julian Szymanski. An iot-based computational framework for healthcare monitoring in mobile environments. *Sensors*, 17(10), 2017.

[101] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 33–43, New York, NY, USA, 1989. ACM.

[102] L. Nastase. Security in the internet of things: A survey on application layer protocols. In *Proc. of 21st International Conference on Control Systems and Computer Science (CSCS)*, pages 659–666, May 2017.

[103] Prabhu Natarajan, Pradeep K. Atrey, and Mohan Kankanhalli. Multi-camera coordination and control in surveillance systems: A survey. *ACM Transaction on Multimedia Computing, Communication and Applications.*, 11(4):57:1–57:30, June 2015.

[104] M. Nawir, A. Amir, N. Yaakob, and O. B. Lynn. Internet of things (iot): Taxonomy of security attacks. In *Proc. of International Conference on Electronic Design (ICED)*, pages 321–326, Aug 2016.

[105] S. K. Nayar. Catadioptric omnidirectional camera. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 482–488, June 1997.

[106] 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*, pages 5326–5329 Vol. 5, May 2005.

[107] K. T. Nguyen, N. Oualha, and M. Laurent. Lightweight certificateless and provably-secure signcryptosystem for the internet of things. In *Proc. IEEE Trustcom/BigDataSE/ISPA*, volume 1, pages 467–474, Aug 2015.

[108] Shufen Niu, Ling Niu, Xiyan Yang, Caifen Wang, and Xiangdong Jia. Heterogeneous hybrid signcryption for multi-message and multi-receiver. *PloS one*, 12(9):e0184407, 2017.

[109] K. Obraczka, R. Manduchi, and J. J. Garcia-Luna-Aveces. Managing the information flow in visual sensor networks. In *Proc. The 5th International Symposium on Wireless Personal Multimedia Communications*, volume 3, pages 1177–1181, Oct 2002.

[110] Christof Paar and Jan Pelzl. *Understanding cryptography: a textbook for students and practitioners.* Springer Science & Business Media, 2009.

[111] 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)*, pages 242–247, Sep 2016.

[112] Z. Pala and N. Inanc. Smart parking applications using rfid technology. In *Proc. 1st Annual RFID Eurasia*, pages 1–3, Sep. 2007.

[113] Liaojun Pang, Xuxia Yan, Huiyang Zhao, Yufei Hu, and Huixian Li. A novel multi-receiver signcryption scheme with complete anonymity. *PloS one*, 11(11):e0166173, 2016.

[114] V. Patchava, H. B. Kandala, and P. R. Babu. A smart home automation technique with raspberry pi using iot. In *Poc. International Conference on Smart Sensors and Systems (IC-SSS)*, pages 1–4, Dec 2015.

[115] A. B. Pawar and S. Ghumbre. A survey on iot applications, security challenges and counter measures. In *Proc. International Conference on Computing, Analytics and Security Trends (CAST)*, pages 294–299, Dec 2016.

[116] M. R. Perbawa, D. I. Afryansyah, and R. F. Sari. Comparison of ecdsa and rsa signature scheme on nlsr performance. In *Proc. IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*, pages 7–11, Nov 2017.

[117] V. Popescu, B. Benes, P. Rosen, J. Cui, and L. Wang. A flexible pinhole camera model for coherent nonuniform sampling. *IEEE Computer Graphics and Applications*, 34(4):30–41, July 2014.

[118] F. Porikli, F. Bremond, S. L. Dockstader, J. Ferryman, A. Hoogs, B. C. Lovell, S. Pankanti, B. Rinner, P. Tu, and P. L. Venetianer. Video surveillance: past, present, and now the future [dsp forum]. *IEEE Signal Processing Magazine*, 30(3):190–198, May 2013.

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

[120] G. Pudics, M. Z. Szab-Resch, and Z. Vmossy. Safe robot navigation using an omnidirectional camera. In *Proc. 16th IEEE International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 227–231, Nov 2015.

[121] Darren Quick and Kim-Kwang Raymond Choo. Digital forensic intelligence: Data subsets and open source intelligence (dfint+osint): A timely and cohesive mix. *Future Generation Computer Systems*, 78:558 – 567, 2018.

[122] Mohammad Rahimi, Rick Baer, Obimdinachi I. Iroezi, Juan C. Garcia, Jay Warrior, Deborah Estrin, and Mani Srivastava. Cyclops: In situ image sensing and interpretation in wireless sensor networks. In *Proc. of the 3rd International Conference on Embedded Networked Sensor Systems*, SenSys '05, pages 192–204, New York, NY, USA, 2005. ACM.

[123] S Ramakrishnan. *Cryptographic and Information Security Approaches for Images and Videos.* CRC Press, 2018.

[124] P. Rashidi and A. Mihailidis. A survey on ambient-assisted living tools for older adults. *IEEE Journal of Biomedical and Health Informatics*, 17(3):579–590, May 2013.

[125] Srivaths Ravi, Anand Raghunathan, and Sunil Kocher, Pauland Hattangady. Security in embedded systems: Design challenges. *ACM Transactions on Embedded Computing Systems (TECS)*, 3(3):461–491, August 2004.

[126] M. Reisslein, B. Rinner, and A. Roy-Chowdhury. Smart camera networks. *Computer*, 47(5):23–25, May 2014.

[127] B. Rinner and T. Winkler. Privacy-protecting smart cameras. In *Proc. of the International Conference on Distributed Smart Cameras*, ICDSC '14, pages 40:1–40:5, New York, 2014. ACM.

[128] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communication. ACM*, 21(2):120–126, February 1978.

[129] J. E. Y. Rosseboe and R. Braek. Towards a framework of authentication and authorization patterns for ensuring availability in service composition. In *Proc. First International Conference on Availability, Reliability and Security (ARES'06)*, pages 10 pp.–215, April 2006.

[130] M. Rusci, D. Rossi, M. Lecca, M. Gottardi, E. Farella, and L. Benini. An event-driven ultra-low-power smart visual sensor. *IEEE Sensors Journal*, 16(13):5344–5353, July 2016.

[131] Manuele Rusci, Davide Rossi, Elisabetta Farella, and Luca Benini. A sub-mw iot-endnode for always-on visual monitoring and smart triggering. *IEEE Internet of Things Journal*, 4:1284–1295, 2017.

[132] Farzad Samie, Lars Bauer, and Jörg Henkel. Iot technologies for embedded computing: A survey. In *Proc. of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, CODES '16, pages 8:1–8:10, New York, NY, USA, 2016. ACM.

[133] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos. Edge analytics in the internet of things. *IEEE Pervasive Computing*, 14(2):24–31, Apr.-June 2015.

[134] Davide Scaramuzza. *Omnidirectional Vision. From Calibration to Root Motion Estimation.* PhD thesis, ETH Zurich, 2007.

[135] Davide Scaramuzza. *Omnidirectional Camera*, pages 552–560. Springer US, Boston, MA, 2014.

[136] C. Scharfenberger, S. Chakraborty, and G. Farber. Robust image processing for an omnidirectional camera-based smart car door. In *Proc. IEEE/ACM/IFIP 7th Workshop on Embedded Systems for Real-Time Multimedia*, pages 106–115, Oct 2009.

[137] M. Schneider and Shih-Fu Chang. A robust content based digital signature for image authentication. In *Proc. International Conference on Image Processing*, volume 3, pages 227–230, Sep 1996.

[138] M. Schwarting, T. Burton, and R. Yampolskiy. On the obfuscation of image sensor fingerprints. In *Proc. Annual Global Online Conference on Information and Computer Technology (GOCICT)*, pages 66–69, Nov 2015.

[139] Seung-Hyun Seo, Jongho Won, and Elisa Bertino. pclsc-tkem: a pairing-free certificateless signcryption-tag key encapsulation mechanism for a privacy-preserving iot. *Transactions on Data Privacy*, 9(2):101–130, 2016.

[140] D. N. Serpanos and A. Papalambrou. Security and privacy in distributed smart cameras. *Proc. of the IEEE*, 96(10):1678–1687, Oct 2008.

[141] Martin Serror, Martin Henze, Sacha Hack, Marko Schuba, and Klaus Wehrle. Towards in-network security for smart homes. In *Proc. of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, pages 18:1–18:8, New York, NY, USA, 2018. ACM.

[142] K. Seyid, V. Popovic, O. Cogal, A. Akin, H. Afshari, A. Schmid, and Y. Leblebici. A real-time multi aperture omnidirectional visual sensor based on an interconnected network of smart cameras. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(2):314–324, Feb 2015.

[143] Z. Shao, J. Cai, and Z. Wang. Smart monitoring cameras driven intelligent processing to big surveillance video data. *IEEE Transactions on Big Data*, 4(1):105–116, March 2018.

[144] M. Sharafi, F. Fotouhi-Ghazvini, M. Shirali, and M. Ghassemian. A low power cryptography solution based on chaos theory in wireless sensor nodes. *IEEE Access*, 7:8737–8753, 2019.

[145] Y. Shi, J. Han, X. Wang, J. Gao, and H. Fan. An obfuscatable aggregatable signcryption scheme for unattended devices in iot systems. *IEEE Internet of Things Journal*, 4(4):1067–1081, Aug 2017.

[146] K. Shim. A survey of public-key cryptographic primitives in wireless sensor networks. *IEEE Communications Surveys Tutorials*, 18(1):577–601, Jan 2016.

[147] Bruno M.C. Silva, Joel J.P.C. Rodrigues, Isabel de la Torre Dez, Miguel Lpez-Coronado, and Kashif Saleem. Mobile-health: A review of current state in 2015. *Journal of Biomedical Informatics*, 56:265–272, 2015.

[148] 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*, volume 6241, pp. 8, Apr 2006.

[149] A. F. Symon, N. Hassan, H. Rashid, I. U. Ahmed, and S. M. T. Reza. Design and development of a smart baby monitoring system based on raspberry pi and pi camera. In *Proc. 4th International Conference on Advances in Electrical Engineering (ICAEE)*, pages 117–122, Sep. 2017.

[150] C. H. Tan. Insider-secure signcryption kem/tag-kem schemes without random oracles. In *Proc. Third International Conference on Availability, Reliability and Security*, pages 1275–1281, March 2008.

[151] S. Thavalengal, R. Vranceanu, R. G. Condorovici, and P. Corcoran. Iris pattern obfuscation in digital images. In *Proc. IEEE International Joint Conference on Biometrics*, pages 1–8, Sep. 2014.

[152] P. Y. Ting, J. L. Tsai, and T. S. Wu. Signcryption method suitable for low-power iot devices ina wireless sensor network. *IEEE Systems Journal*, pages 1–10, 2017.

[153] S. Ullah, L. Marcenaro, and B. Rinner. Secure smart cameras by aggregate-signcryption with decryption fairness for multi-receiver iot applications. *Sensors*, 19(2), 2019.

[154] S. Ullah, B. Rinner, and L. Marcenaro. Smart cameras with onboard signcryption for securing iot applications. In *Proc. IEEE Global Internet of Things Summit (GIoTS)*, pages 1–6, June 2017.

[155] S. Ullah, F. Russo, L. Marcenaro, and B. Rinner. Aggregate-signcryption for securing smart camera iot applications. In *Proc. IEEE Global Internet of Things Summit (GIoTS)*, pages 1–6, June 2018.

[156] A. Upadhyaya, V. Shokeen, and G. Srivastava. Image encryption: Using aes, feature extraction and random no. generation. In *Proc. International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*, pages 1–4, Sep. 2015.

[157] Arezoo Vejdanparast, Peter R. Lewis, and Lukas Esterle. Online zoom selection approaches for coverage redundancy in visual sensor networks. In *Proc. 12th International Conference on Distributed Smart Cameras*, ICDSC, pages 15:1–15:6, New York, NY, USA, 2018. ACM.

[158] Giuseppe Veneziano. Video camera device and method to monitor a child in a vehicle, jan 2019. US Patent App. 10/178,357.

[159] V. P. Venkatesan, C. P. Devi, and M. Sivaranjani. Design of a smart gateway solution based on the exploration of specific challenges in iot. In *Proc. International Conference on IoT in Social, Mobile, Analytics and Cloud (I-SMAC)*, pages 22–31, Feb 2017.

[160] Hongwei Wang and Wenbo He. A reservation-based smart parking system. In *Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 690–695, April 2011.

[161] M. Wang, C. Huang, and H. Lin. An intelligent surveillance system based on an omnidirectional vision sensor. In *Proc. IEEE Conference on Cybernetics and Intelligent Systems*, pages 1–6, June 2006.

[162] X. Wang, A. Chowdhery, and M. Chiang. Networked drone cameras for sports streaming. In *Proc. IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 308–318, June 2017.

[163] Y. Wang and T. Li. Study on digital watermarking algorithm based on wavelet transform and chaotic encryption. In *Proc. International Conference on Electrical and Control Engineering*, pages 853–855, Sep. 2011.

[164] W. H. Widen. Smart cameras and the right to privacy. *Proceedings of the IEEE*, 96(10):1688–1697, Oct 2008.

[165] E. K. Win, T. Yoshihisa, Y. Ishi, T. Kawakami, Y. Teranishi, and S. Shimojo. A lightweight multi-receiver encryption scheme with mutual authentication. In *Proc. IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 491–497, July 2017.

[166] Ei Khaing Win, Tomoki Yoshihisa, Yoshimasa Ishi, Tomoya Kawakami, Yuuichi Teranishi, and Shinji Shimojo. Lightweight and secure certificateless multi-receiver encryption based on ecc. *Journal of Information Processing*, 26:612–624, 2018.

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

[168] T. Winkler and B. Rinner. Trustcam: Security and privacy-protection for an embedded smart camera based on trusted computing. In *Proc. 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 593–600, Aug 2010.

[169] T. Winkler and B. Rinner. Sensor-level security and privacy protection by embedding video content analysis. In *Proc. 18th International Conference on Digital Signal Processing (DSP)*, pages 1–6, July 2013.

[170] T. Winkler and B. Rinner. Security and privacy protection in visual sensor networks: A survey. *ACM Computing Surveys(CSUR).*, 47(1):2:1–2:42, May 2014.

[171] 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)*, pages 1–6, Apr 2015.

[172] Thomas Winkler and Bernhard Rinner. Demo: Trusteye.m4 – a novel platform for secure visual sensor network applications. In *Proc. of the International Conference on Distributed Smart Cameras*, ICDSC '14, pages 45:1–45:3, New York, USA, 2014. ACM.

[173] W. Wolf, B. Ozer, and T. Lv. Smart cameras as embedded systems. *Computer*, 35(9):48–53, Sep 2002.

[174] J. Won, S. H. Seo, and E. Bertino. Certificateless cryptographic protocols for efficient drone-based smart city applications. *IEEE Access*, 5:3721–3749, 2017.

[175] Ping Wah Wong. A public key watermark for image verification and authentication. In *Proc. International Conference on Image Processing*, volume 1, pages 455–459, Oct 1998.

[176] Hu Xiong, Zhen Qin, and Athanasios V. Vasilakos. *Introduction to Certificateless Cryptography.* CRC Press, Inc., Boca Raton, FL, USA, 2016.

[177] G. Xu, Y. Cao, Y. Ren, X. Li, and Z. Feng. Network security situation awareness based on semantic ontology and user-defined rules for internet of things. *IEEE Access*, 5:21046–21056, 2017.

[178] Song Y Yan. *Number theory for computing.* Springer Science & Business Media, 2002.

[179] M. Yang, N. Bourbakis, and Shujun Li. Data-image-video encryption. *IEEE Potentials*, 23(3):28–34, Aug 2004.

[180] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao. A survey on security and privacy issues in internet-of-things. *IEEE Internet of Things Journal*, 4(5):1250–1258, Oct 2017.

[181] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, and Tetsuya Izu. Computational hardness of ifp and ecdlp. *Applicable Algebra in Engineering, Communication and Computing*, 27(6):493–521, 2016.

[182] F. Ye, Y. Qian, and R. Q. Hu. Smart service-aware wireless mixed-area networks. *IEEE Network*, 33(1):84–91, January 2019.

[183] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32, Feb 2014.

[184] H. Zang, L. Min, and L. Cao. An image encryption and digital signature scheme based on generalized synchronization theorem. In *Proc. International Conference on Computational Intelligence and Security*, volume 1, pages 504–510, Dec 2009.

[185] Ali Akbar Zarezadeh, Christophe Bobda, Franck Yonga, and Michael Mefenza. Efficient network clustering for traffic reduction in embedded smart camera networks. *Journal of Real-Time Image Processing*, 12(4):813–826, Dec 2016.

[186] Cha Zhang and Tsuhan Chen. A self-reconfigurable camera array. In *ACM SIGGRAPH 2004 Sketches*, SIGGRAPH '04, pages 243–254, New York, NY, USA, 2004. ACM.

[187] Meiyan Zhang and Wenyu Cai. Vision mesh: A novel video sensor networks platform for water conservancy engineering. In *Proc. 3rd International Conference on Computer Science and Information Technology*, volume 4, pages 106–109, July 2010.

[188] Wenyu Zhang, Zhenjiang Zhang, Dapeng Qi, and Yun Liu. Automatic crack detection and classification method forsubway tunnel safety monitoring. *Sensors*, 14(10):19307–19328, 2014.

[189] Xiao Zheng and X. Li. An efficient certificateless signcryption in the standard model. In *Proc. IEEE International Conference on Cloud Computingand Big Data Analysis (ICCCBDA)*, pages 199–205, July 2016.

[190] Y. Zheng. Signcryption and its applications in efficient public key solutions. In Eiji Okamoto, George Davida, and Masahiro Mambo, editors, *Information Security*, volume 1396 of *Lecture Notes in Computer Science*, pages 291–312. Springer Berlin Heidelberg, 1998.

[191] Yuliang Zheng. Digital signcryption or how to achieve cost(signature & encryption) cost(signature) + cost(encryption). In Burton S. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, pages 165–179, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

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

[193] Xuanwu Zhou, Zhigang Jin, Yan Fu, Huaiwei Zhou, and Lianmin Qin. Short signcryption scheme for the internet of things. *Informatica*, 35(4), 2011.