

Anonymous Access Monitoring of Indoor Areas

SERENA NICOLAZZO¹, ANTONINO NOCERA², AND DOMENICO URSINO³

¹Polytechnic University of Marche, 60131 Ancona, Italy

²Department of Electrical, Computer and Biomedical Engineering, University of Pavia, 27100 Pavia, Italy

³DII, Polytechnic University of Marche, 60121 Ancona, Italy

Corresponding author: Antonino Nocera (antonino.nocera@unipv.it)

ABSTRACT Video surveillance of public spaces is a feature of modern society that has expanded quite quickly and in a pervasive way during the last decades becoming a fundamental need for both individual and collective security. But, as the sophistication of this type of systems increases, the concern about threat to individuals' right of privacy raises as well. Indeed, the video surveillance systems could breach personal privacy because location is clearly one of the most sensitive people information. Hence, preserving location privacy while achieving utility from it, is a challenging problem demanding the investigation of researchers. This paper tackles this non-trivial issue by designing a novel privacy-preserving architecture able to anonymously monitoring people access at the entrance of critical areas in an indoor space. At the same time our approach is able to provide full accountability in case of an accident or a legal requirement. Interestingly, our protocol is robust to server-side attacks and is efficient enough to be applied indoors through a set of IoT (Internet of Things) smart camera devices.

INDEX TERMS Anonymous access, privacy, physical access monitoring, Internet of Things.

I. INTRODUCTION

In the last years, there has been a tremendous increase of video surveillance systems in public locations, both indoors and outdoors, such as stores, museums, stations, schools, and airports, with the objective of reducing crimes and maintain public order [1]–[4]. These monitoring systems track users continuously and generate an enormous amount of potentially sensitive information, which represents an intrinsic threat for people's privacy [5]–[7].

Recently, these technologies are evolving towards the so called Internet of Multimedia Things (hereafter, IoMT), a novel paradigm defined as a network of smart devices requiring higher computational capabilities than classical sensors belonging to the Internet of Things (hereafter, IoT) [8]–[10]. IoT is the network model filling the gap between the cyber and physical world. It is able to connect pervasive objects around us. However, when it comes to surveillance systems, network dynamism, higher throughput, and the necessity to exchange big and heterogeneous data require the revision and amplification of the traditional scalar IoT system. The scientific literature refers to such a revision as IoMT.

The associate editor coordinating the review of this manuscript and approving it for publication was Yogachandran Rahulamathavan.

In general, the diffusion of video surveillance solutions, seen as a particular aspect of IoMT, has been quite tolerated and even accepted, because of the perceived benefits in terms of public safety. However, the potential individuals' privacy invasion, especially in indoor environments, has always raised important concerns. The perception of compromised privacy is particularly strong for technologies which, by default, keep a visual record of people's actions [11], [12].

This issue became even more evident with the diffusion of cloud-based solutions allowing for the capillary availability of such a security technology. Indeed, in this context, video surveillance systems are provided under a business model of Video Surveillance as a Service (VSaaS, for short) [13]. This model provides the distribution of storage space, infrastructures and computational power, necessary for handling large amounts of video data in the cloud. According to this strategy, even a small company can monitor its physical spaces exploiting remote server capabilities as a common Infrastructure as a Service (IaaS, for short). In practice, the client just installs some smart cameras and relies on the power of the cloud for computing and storing capabilities. This raises new challenges in term of privacy, because even the cloud provider could be honest-but-curious.

Still in this context, one of the most diffuse application of video surveillance and IoMT is physical access monitoring. In this case, the video acquisition devices are capable of

detecting human faces to build access logs [14]–[16]. These logs are then stored in dedicated servers, even dislocated in the cloud, to enable a-posteriori analysis of accesses in case of accidents. During this analysis, logs could be used for identification; in particular, face recognition software can automatically identify people from face snapshots leveraging known sources, such as databases of drivers' license photos, and thereby track their movements inside the monitored environment.

However, though this security solution is essential and extremely important in case of accidents for accountability, the access logs stored in the dedicated servers expose people to privacy threats as they could be 1 continuously tracked, even by honest-but-curious providers, also in the absence of suspicious events. To face this issue, a-posteriori analysis of such logs should be allowed only to law enforcement agencies; however, the mere presence of accessible repositories of video frames containing people's faces represents a critical point for privacy protection.

To give a better understanding of this issue, consider a possible real-life application scenario in which an access monitoring system based on a video surveillance solution is used to protect the physical entrances (both outdoor and indoor) of a university, an airport or a bank. Logs generated by such a system could be used to identify suspects in the event of thefts or terrorist attacks. More recently, due to the outbreak of the COVID-19 pandemic disease [17], these solutions have been more and more adopted to derive the network of physical contacts for infected persons. Typically, in these kind of systems, data is traced without any care of privacy or anonymity. Moreover, detailed logs are maintained with precise timestamp and location information, hence a complete track of people movements inside areas can be derived. In such a scenario, it is evident that not only the data about people's face snapshots has to be protected, but also the position in which each snapshot has been captured along with its timestamp.

Of course, to address this problem, solutions based on the use of cryptography (typically homomorphic encryption, to enable some form of computation on encrypted data) could be adopted [18], [19]. Such solutions can prevent from unauthorized access to logs also in case of server-side attacks coming from external malicious entities. However, they cannot guarantee protection against server-side attacks involving anomalous behavior of the system owner himself. This can happen in scenarios in which either the system owner cannot be assumed to be a trusted party or he can be considered honest-but-curious. Indeed, data encryption should be performed by the system owner, who, therefore, must be aware of all the technicalities behind it (including the knowledge of cryptographic keys). Hence, he can have clear access to logs, thus causing privacy leakages.

Enabling a protection to server-side attacks in such a context would require a more refined security strategy, which does not leverage only a single party. Adopting a distributed solution would imply a non-collusion assumption among

involved parties, with the exception of particular critical circumstances as prompted by law enforcement agencies. Such strategy should provide a mechanism to allow for full accountability and also include the possibility of isolating the portion of data of interest for investigation, without exposing all the dataset to privacy threats.

In this paper, we give a contribution in this setting by designing a distributed solution for private-preserving access monitoring systems based on video surveillance. The proposed solution protects against server-side attacks to privacy also in case the system owner is involved. Moreover, it retains the ability to reconstruct the original data, thus providing full accountability, in case of a legal requirement.

Our proposal makes use of a network of smart cameras located at the entrance of monitored zones to acquire people's face snapshots. An ad-hoc image secret sharing algorithm is adopted to conceal the identity of the person in each snapshot. Sharing image secretly has acquired great interest in modern cryptography for both commercial and military applications [20]. The basic idea underlying this approach is to decompose an image into several shards so that only by combining data from a minimum subset of them, the original image can be re-built. In our approach, smart cameras decompose the acquired snapshot into secret shards and send them to at least two separated and non-colluding servers. Only a subset of them can be held and managed by the system owner himself, whereas the others are provided by a trusted-third party. Using a P2P network powered with an Onion-based routing protocol, the smart cameras cooperate to anonymously send the shards to the servers in such a way as to protect the source of each snapshot, and therefore the information about the specific entrance.

Our solution protects against server-side attacks because the data stored in either the system owner servers or the trusted-third party ones are not sufficient alone to re-identify the subject or to track him.

The plan of this paper is as follows. In Section II, we examine related literature, whereas the background about image secret sharing and onion protocol is outlined in Section III. In Section IV, we present the proposed model in detail, whereas our solution is described in Section V. Section VI is devoted to the analysis of the security properties of our solution. Then, an extensive and thorough experimental campaign to evaluate the performance of our approach is presented in Section VII. In Section VIII we discuss the main limitations of our approach. Finally, in Section IX, we draw our conclusions and have a look at possible future developments of our research efforts.

II. RELATED WORKS

A lot of research has been done in the context of IoMT in order to propose novel security systems based on multimedia data, e.g., retina or biometric scanning, voice recognition, and video surveillance systems [10]. In particular, video surveillance devices have become a key requirement for smart city application in order to ensure public safety for crime

detection, residential or assisted living facilities [21], [22], industrial surveillance, and other critical spaces [23], [24]. In [21], an IoT residential surveillance system is designed. The proposed approach is based on an Android application and some hardware sensors, able to detect suspicious movements within the specified range. The scheme described in [24] is useful for content-based retrieval of visual data to achieve passenger safety in a public transport system. This approach leverages a natural language processing module along with an ontology and an image analysis based video surveillance.

However, this ubiquitous use of IoMT devices, especially for video surveillance, can lead to the perception of privacy loss and security issues from the user point of view [25]. For this reason, currently, many researchers working in this field have focused their studies on approaches to minimize privacy loss in this scenario.

Basically, we can divide research efforts in two main lines, namely: (i) solutions dealing with computer vision approaches for video surveillance [26]–[29], and (ii) solutions providing anonymity that leverages communication protocols, like mix routing [30].

The aim of the first group of solutions is representing just enough of the information contained in a video stream in order to allow video-based tasks, but, at the same time, hiding people identity, or others details containing sensitive information. In particular, in [26] the authors present a new camera with onboard processing that produces a video stream with the privacy-intrusive information already removed. The work presented in [29] deals with a privacy-preserving vision system for estimating the size of not homogeneous crowds, which does not depend on object detection or feature tracking. The approach proposed in [27] consists in parsing the codestream and in pseudo-randomly inverting some of the bits corresponding to the regions of interest of a MPEG-4 video stream. In both the transform-domain and code-stream domain approaches, the scrambling process depends on a secret encryption key, which can be in possession of law-enforcement authorities. Instead, the authors of [28] present a method to de-identify faces digitally modifying images, so that a face recognition software cannot reliably relate people to their captured images. In the systems presented in [31]–[34], the authors choose to obfuscate the complete humans silhouettes [31], [32] or the bounding boxes covering a whole human body [33], [34] in order to protect privacy.

The main drawback of these algorithms, based on computer vision, is that even if they have high performances in objects detection, they cannot guarantee anonymity. Indeed, contextual information may be enough to uniquely identify a person even when all identifying characteristics are obscured in the video. This is enough to lead to identity leakage and, consequently, privacy loss [35], [36]. In our approach, we do not use computer vision algorithms to alter the original image or part of it. On the contrary, we produce K layers (or shards) from an original image, such that layer by itself has any useful

information, but the layers in the whole retain the original image.

An approach similar to ours, always in the context of computer vision, is presented in [37], where the authors describe a framework to carry out privacy preserving surveillance. Although also this approach is based on Secret Sharing paradigm, it has quite a different focus with respect to ours. Indeed, we present a complete architecture to provide anonymous surveillance, but neither the servers nor any other nodes perform computations on the shards. Only in case of legal requirement the original image will be reconstructed. We detailed the discussion about this paper in the next section.

The approach presented in [37] is based on the concept of homomorphic encryption [38] that allows direct computation on encrypted data, without access to the secret key. A common use of homomorphic encryption happens when a data owner wants to leverage cloud for processing and saving his data, but the cloud service provider is not trusted. Using a homomorphic encryption scheme, data are encrypted and sent to the server. This can perform computations on the cypher data and send the results to the data owner without a decryption step. The latter is the only one able to decrypt data, since he alone has the secret key. Applying homomorphic encryption in our approach would have meant not using the image secret sharing scheme based on the Shamir algorithm. A possible application of such an encryption in our scenario could be done as follows:

- 1) the camera encrypts the message with homomorphic encryption scheme.
- 2) it sends the message to the *TTP Server*.

Although this solution can prevent an unauthorized person from revealing the identity of people in the logs, it would not protect against server-side attacks involving the system owner. In other words, the keys holder can access the logs. By contrast, in our architecture, in case of legal requirements, only the cooperation of two entities could reveal people identity.

The second group of algorithms based on communication protocols comprises all those schemes for anonymous service usage. According to these systems, the user identifier can be omitted and the network routing is addressed by mechanisms such as Crowds [39] or Onion Routing [40], [41], which provide sender anonymity. For instance, in [42], the authors introduce the concept of mix zones in order to protect the privacy of user location information. In particular, the ability to prevent other parties from learning one's current or past location is provided through the frequently change of pseudonyms rather than by employing the real user identities inside connected spatial regions.

The use of pseudonyms is a recurrent approach to solve the issue of privacy preserving in indoor location system [23], [43], [44]. For instance, the Active Badge system [43] detects the location of each user and broadcasts this information to everyone in the building. The main limitation of this system, as originally deployed, is that it provides no mechanisms

to limit the dissemination of information about individuals' location. This system was then modified by Ian W. Jackson to address this issue [44]. In this modified version, a badge does not reveal its identity to the sensor detecting its position, but only to a trusted-party at the network edge. Moreover, through encrypted and anonymized communication, the entity performing traffic observation does not reveal which computer a given badge trusts. The badge's owner can then use traditional access control methods to allow, or disallow, other entities to query the badge's location.

In [23], the authors propose an RFID-based technique to trace people. The described approach preserves privacy introducing a certain degree of uncertainty with by implementing a k -anonymity property. For this paradigm a user who accessed a place, at a given time, is identified with probability $1/k$. Although our scenario tackles the problem of surveillance in indoor location systems, it does not require any personal equipment and this allows a more flexible and scalable adoption of our solution. Indeed, we leverage the use of snapshots to identify people.

Over time, a wide range of application scenarios in the context of Location Based System (LBS, for short) took advantage of a number of techniques to guarantee privacy. Just to name a few of them, we cite: (i) location cloaking [30], [42], [45], that aims at perturbing location data by introducing random noise in order to guarantee user's privacy; (ii) obfuscation or generation of fake contextual data [46], [47]; and, (iii) k -anonymity, which requires that location information inside a message sent from a user to a LBS should be indistinguishable from at least k other messages coming from different mobile nodes [23], [48].

LBS are software services which exploit geographic data and information to provide different services; they are very distant from our application scenario. Indeed, our approach leverages snapshots to identify people, and this kind of assets cannot be handled by the above cited techniques.

In the next section, we present a deeper comparison of our approach with the state-of-the-art schemes for privacy-preserving distributed surveillance.

A. COMPARATIVE ANALYSIS WITH STATE-OF-THE-ART SCHEMES FOR DISTRIBUTED SURVEILLANCE CAMERAS

In this section, we compare our approach with some related works operating in the context of privacy-preserving monitoring systems. In particular, we take the following properties into consideration:

- **Identity Privacy**, that is the capability of protecting end users from identity privacy loss, while monitoring people and environment.
- **Access Location Privacy**, that is the capability of protecting end users from the loss of their location privacy, and simultaneously monitoring people and environment.
- **A-posteriori Accountability**, that is the possibility to reconstruct the identity of a person who accessed a certain zone, in case of an accident or a legal requirement.

TABLE 1. Comparison of our approach with the state-of-the-art ones.

Article	Identity Privacy	Access Location Privacy	A-posteriori Accountability	Server-Side attacks robustness
Ours	x	x	x	x
[37]	x	-	x	-
[28]	x	-	-	-
[50]	x	-	-	-
[27]	x	-	-	-
[51]	x	-	-	-
[52]	x	-	x	-
[53]	x	-	x	-
[54]	x	x	-	x

- **Server-side attack robustness**, that is the system resilience to server-side attacks also performed by the server owner, who could be malicious or simply honest-but-curious.

In Table 1, all the works we refer in this section are listed altogether with the above properties. The symbol 'x' denotes that the cited paper provides the corresponding property.

The approach presented in [37] aims at enabling distributed secure processing and images storage, retaining the possibility to reconstruct the original data in case of a legal requirement. In this approach, a camera computes N shards from an image and sends them to a number of independent servers, each performing some basic operations on its own shard. At the end, an observer puts together the results of these basic operations to get the final outcome as it would be obtained from the original image. This paper describes a framework to carry out privacy preserving surveillance leveraging a Secret Sharing paradigm. It is able to reconstruct the original data (a-posteriori accountability), but it cannot protect against server-side and network attacks. Indeed the independent servers know the source camera, which the shard is coming from, and the corresponding timestamp. Therefore, the protection of access location privacy is not guaranteed in case of malicious access to these servers. In our approach, the use of an ad-hoc scheme for package delivery, which borrows some concepts from mix-nets [49] and leverages an Onion routing protocol, can conceal the source camera, thus providing protection of information about access location. Furthermore cameras add random delays to input packets before forwarding them. In this way, our approach can successfully prevent also attacks based on traffic and access time analysis.

The paper presented in [28] describes a privacy-enabling algorithm, named k-Same, able to obscure the face of a person. The de-identified faces cannot be recognized by an automatic face recognition software, even though many facial details are preserved. In this way, a person can be automatically detected in an image, but he cannot be correctly recognized. Similar approaches are presented in [27], [50], [51]. In particular [51] leverages warping techniques (common for animation and artistic purposes) to obfuscate faces in video surveillance. These systems do not provide

any guarantee of location privacy, because only the face of the person entering a location is obscured. Information about the event timestamp and the accessed area are not protected. On the other hand, the link between the access event and the involved person is not preserved. Moreover, the original video is not stored; therefore, an a-posteriori accountability verification is not possible. Furthermore, the server dedicated to the image processing task knows the source camera and the snapshot timestamp. As a consequence, these algorithms do not provide protection against server-side attacks.

The authors of [52], [53] present two similar approaches for IoMT based on Region of Interest (RoI) privacy protection systems. Through these approaches, only a limited number of privacy-sensitive areas (ROIs) is encrypted in the video independently and synchronously. The encrypted ROIs are stored at the camera side and can be accessed by authenticated users. The surveillance video without ROIs can be watched in real-time by any user online. Although a-posteriori accountability is guaranteed, this system cannot provide location privacy. Moreover, if the owner of the server storing the encrypted ROIs is malicious or honest-but-curious, he can access the ROIs, the source cameras of the video and the related timestamps.

Finally, the paper proposed in [54] deals with a blockchain-based privacy protection scheme for video surveillance suitable for IoMT, called Lib-Pri. It guarantees people monitoring by capturing their image in real-time videos, without compromising their identity or location privacy. Lib-Pri leverages a federated blockchain network capable of carrying out integrity checking, blurring keys management, feature sharing, and video access sanctioning. This approach is fully distributed, and, hence is resilient to server-side attacks. However, its main drawback is that it cannot perform a-posteriori accountability in case of an accident or legal requirement, because the original videos are not maintained in a storage.

III. BACKGROUND

This section we present the background context needed to understand our approach. In particular, in Section III-A, we describe the original Secret Sharing Scheme and, then, we focus on Image Secret Sharing scheme in a new version able to share an image as a secret. After that, in Section III-B, we describe the basics of the Onion Routing protocol exploited in our solution.

A. SECRET SHARING SCHEME

The problem of Secret Sharing Scheme (SSS, for short) was first addressed in 1979 by Blakley [55] and Shamir [56]. The approach followed was to distribute the encryption/decryption key to a number n of participants so that any $k < n$ participants can reconstruct the secret, and any $(k - 1)$ or less participants cannot reconstruct it. These two approaches work as follows. First, a secret number K is divided into n shadows (K_1, \dots, K_n) . To perform such a split, the approach randomly selects a prime number p and a $r - 1$

degree polynomial (see Equation 1).

$$q(x) = (a_0 + a_1x + \dots + a_{r-1}x_{r-1}) \bmod p; \quad (1)$$

where $K_0 = K = a_0$, $K_1 = q(1) = a_0 + a_1$ and $K_i = q(i)$, and the integer coefficients (a_0, \dots, a_n) are chosen in the interval $[0, p - 1]$. Observe that each K_i represents a shadow. Given any r pairs of these n pairs (i, K_i) with $i \in [1, n]$, we can find the coefficients a_0, \dots, a_{r-1} of $q(x)$ by the Lagrange's interpolation, and solve the linear system revealing the secret data $K = a_0$.

Similarly, Shamir's (r, n) threshold scheme has been used to share also a secret image. To do so, an image is split in n frames or shadow images, and any r shadow images (with $(r \ll n)$) of them can be used to restore the whole secret image.

The direct application of Shamir's method implies the following steps:

- consider $K_0 = K = a_0$ as the gray value of the first pixel;
- obtain the corresponding output $q(1), \dots, q(n)$ from Equation 1;
- repeat this process until all pixels of the secret image are processed;
- share (i, K_i) , with $i \in [1, n]$, with n participants.

As a consequence of the above steps, each shadow image has the same size of the secret source image.

In our approach we use a method called Secret Image Sharing [20], leveraging such a logic but with some enhancements.

In particular, because the gray value of a pixel ranges in the interval $[0, 255]$, the authors of [20] let the prime number p be 251 (which is the greatest prime number not higher than 255). To apply the method, we must truncate all the gray values so that they fall in the range $[0, 250]$. The second step is to use a key to generate a sequence to permute the pixels of the secret image in order to increase the security. Indeed, the pixel values in a real picture are not entirely random, because neighboring pixels often have equal or close values. This creates the possibility that one image secret share may be used to recover the secret image by assuming that neighboring pixels have similar or equal values in the first order polynomial function. After the permutation step, the image is divided into several sections of r pixels. For each section the following $r - 1$ degree polynomial is defined:

$$q_j(x) = (a_0 + a_1x + \dots + a_{r-1}x_{r-1}) \bmod 251; \quad (2)$$

Here a_0, a_1, \dots, a_{r-1} are the r pixels of a section j . The n output pixels $q_j(1), \dots, q_j(n)$ of j are sequentially assigned to the n shadow images. Hence, in this case, the size of each shadow image is $1/r$ of the secret image. The last two steps are repeated until all pixels of the permuted image are processed.

Figure 1 shows a numeric example of $(2, 3)$ Secret Image Sharing, where $r = 2$ and $n = 3$.

Following this approach, for starters, all the pixels of an image are permuted with a key v owned by a camera. After

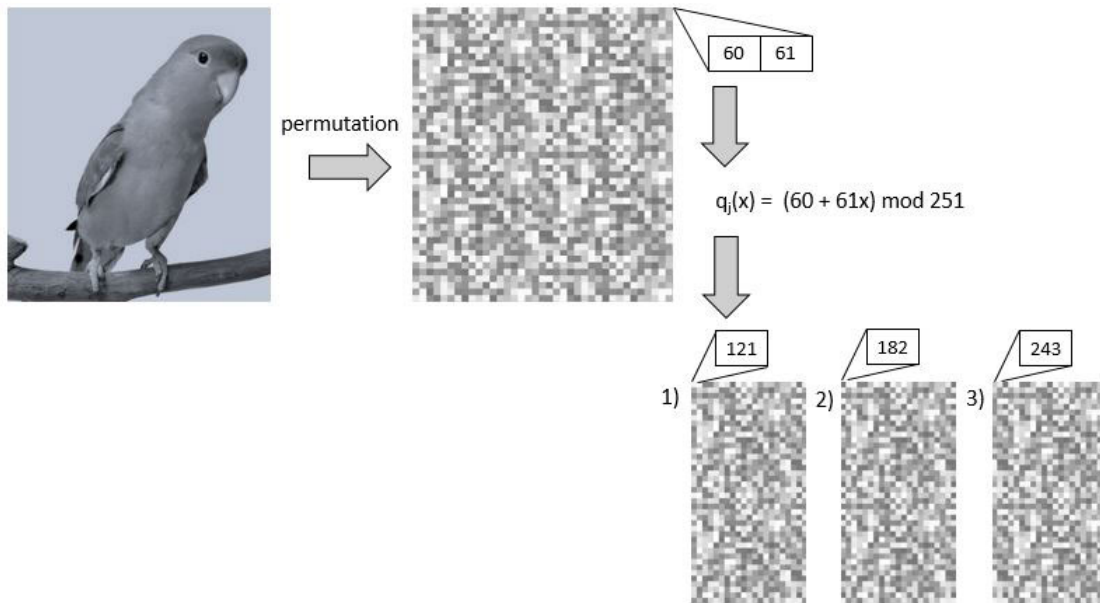


FIGURE 1. Numeric example of (2, 3) secret Image sharing.

this step, the image is divided in sections of 2 pixels. For a section $j = 1$, a first order polynomial function is generated as:

$$q_1(x) = (60 + 61x) \bmod 251 \tag{3}$$

where 60 and 61 are the first two pixel values in the image belonging to section $j = 1$. For our three participants, three shares are computed as (1, 121), (2, 182) and (3, 243). They become the first pixels in three image shares. This process continues until all the pixels are encoded and, at the end, three image shares with half the size of the original image are created. None of the image shares appear to reveal information about the secret image, but any 2 image shares together can be used to reconstruct every pixel value of the original image.

In [20], a lossless secret image sharing method is also presented. In any case, the technical details of the adopted secret image sharing algorithm are orthogonal to our approach and it is possible to adopt either Thien and Lin’s Image Secret Sharing Scheme or other similar approaches, such as, for instance, the ones described in [57], [58].

B. ONION ROUTING

Onion routing [41] is a paradigm useful for anonymous communication over a network. According to this protocol, messages are encapsulated in layers of encryption, similarly to the layers of an onion (as shown in Figure 2). The encrypted message is then routed through the network and, at every hop, each node deciphers with its private key a single layer, uncovering the message’s next destination. When the final layer is decrypted, the message reaches its destination.

In our approach we reuse this onion message structure as a mechanism to achieve sender anonymity.

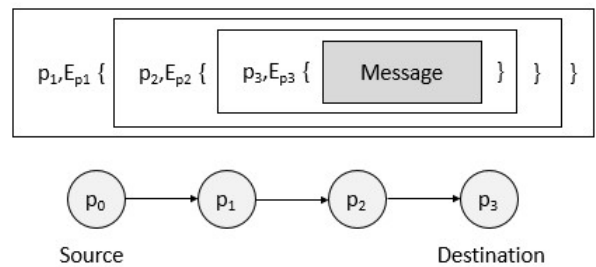


FIGURE 2. Typical message in Onion routing.

IV. PROPOSED MODEL

In this section, we describe the model adopted in our approach to represent all the involved entities. Preliminarily, to improve the readability of this paper, we start by reporting in Table 2 the notations and abbreviations used in the remaining of this paper.

The entities involved in our model are:

- The organization O , which is the owner of the surveillance system.
- The Trusted-Third Party, TTP for short, which is an external organization and observes a non-collusion policy with O , unless explicitly requested by a law enforcement agency.
- The set of users, say U , who access the monitored areas of our environment.
- The set of smart cameras, namely C , which are mounted at each entrance of the monitored areas of our environment. The smart cameras are the main component of our architecture. In this paper, we are making explicit reference to smart cameras equipped with an operating system, network capabilities and enough

TABLE 2. Notations used in this paper.

Symbol	Description
$IoMT$	Internet of Multimedia Thing
U	User who accesses a given area
C_i	Camera of room i belonging to the P2P network
$ C $	Number of smart cameras in the building
$TTPS$	Trusted Third Party Server
HS	Home Server
N	Number of layers
K	Number of shards for identity detection
$E_{C_i}(x)$	The encryption of message x with the key of a camera C_i
$D_{C_i}(x)$	The decryption of message x with the key of an entity C_i
m	message
\tilde{m}	Shuffled message
l_m	Length of the message
sn	Snapshot
ts_{sn}	Timestamp of sn
s	Initial seed
cs	Camera seed
pn_{HS}	Public nonce generated by HS
T_ID	Transaction identifier
$d_{\tilde{m}}$	Digest
M_{cs}	Set containing the mapping between each c_i and the corresponding internal seed cs_i

computational capabilities to perform simple manipulations on captured images. Each smart camera is located at the entrance of the areas to be protected. Additionally, each smart camera is identified by an ordinal number from 0 to $|C|$.

- The P2P Network of Cameras, called *P2P*, is the communication channel adopted by smart cameras to communicate with each other and with the servers. It provides secure point to point communication among system actors.
- The *Home Sever*, *HS* for short, is devoted to store and provide access to all the information related to the monitoring activity of the company. It can be either an on premise server or a cloud entity.
- The Trusted-Third Party Server, *TTPS* for short, is an external component to the organization and is in charge of storing the information sent by the smart cameras. In controlled cases, it may collude with the Organization Server to restore information for accountability. This task is described in Section V-E.

In the next sections, we will describe in detail the behavior of each of the actors, as well as their interactions according to the proposed scheme.

V. THE ANONYMOUS ACCESS MONITORING SCHEME

In this section, we describe our approach for building an anonymous access monitoring scheme in an environment monitored by smart cameras.

A. GENERAL OVERVIEW OF THE PROPOSED SCHEME

As stated in the Introduction, our approach focuses on the protection of user privacy in a scenario characterized by the presence of an access monitoring mechanism based on a

video surveillance system. The idea underlying our solution is that a monitored environment, organized in separated zones, is equipped with smart cameras (i.e., cameras with an operating system, basic computation and networking capabilities) at the entrances of each zone. The smart cameras are configured in such a way as to capture face snapshots of each person trespassing the entrance they monitor. Face snapshots must be preserved to allow for a-posteriori analysis in case of an accident in one of the monitored zones.

For this purpose, each record has to contain the snapshot itself, the identifier of the specific smart camera (and, hence, its position in the environment) that captured the snapshot, and the timestamp of the capturing event.

These records contain information that, if accessed by unintended users, can cause severe violations to people's privacy. Indeed, even an honest-but-curious administrator of the storage infrastructure could represent a threat in this setting. As said in the previous sections, several related works proposed the exploitation of cryptography to address these types of issues by inhibiting the direct access to stored information. However, in our application context, even the storage infrastructure, as well as its administrators, cannot be considered secure and, hence, cryptography-based strategies seem not adequate because the system administrators have still access to both the encrypted content and the keys to decrypt it.

For these reasons, the idea underlying our approach is to split all the information necessary to provide accountability in case of an accident and store them in separated storage infrastructures. In particular, in our scenario we consider the storage server of the entity owner, called *Home Server*, of the surveillance system and leverage a trusted-third party, called *TTP Server*, to store a piece of information.

A general architecture of our approach is reported in Figure 3.

From the analysis of this figure, we can see that our approach consists of the phases reported below.

1) PHASE 1. CAPTURING SNAPSHOT

At the entrance of each monitored zone, the smart cameras capture snapshots of users accessing them. Smart cameras are suitably located to ease the acquisition of images of users' physiology.

2) PHASE 2. CREATING SECRET IMAGES

These images are split into three shards by means of a modified version of the Thien and Lin scheme [20] (see Section III for background details about it). Two of these shards are intended to be stored separately in the two storage servers (the *Home Server* and the *TTP Server*). However, to provide accountability, also the identifier of the smart camera (and, hence, its position) and the timestamp of the snapshot must be stored. As for the timestamp, we can safely store it in one of the two servers, whereas the information about the position in which the snapshot has been taken is a more sensible data. Indeed, the knowledge of the position of the camera, along with the timestamp, allows for attacks based on the possible,

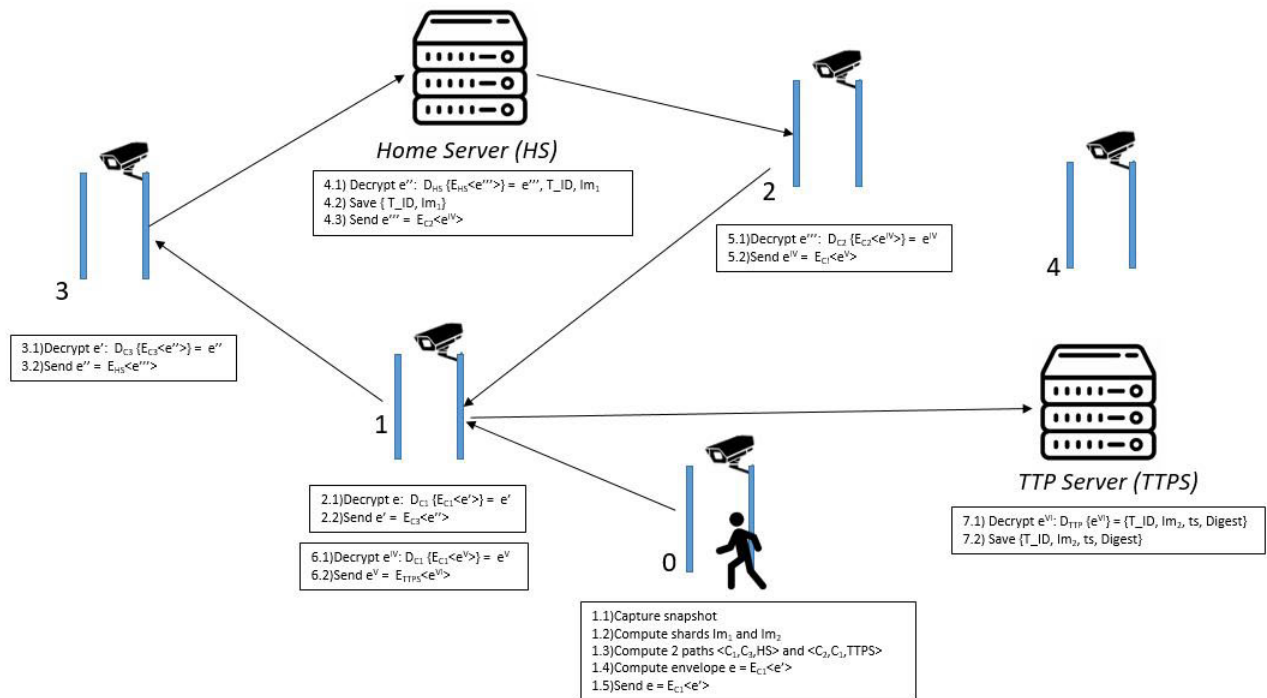


FIGURE 3. General architecture of our approach.

even minimal, background knowledge of the building and the habits of its users [59].

3) PHASE 3. BUILDING THE ENVELOPE

Protecting the information about the position requires the source of the data (i.e., a specific smart camera) to be concealed. For this reason, our approach assumes that all the smart cameras and the servers take part to a P2P network in which message exchange takes place leveraging an Onion-based anonymous communication protocol (see Section III for background information about this protocol). Under this assumption, once a smart camera has acquired a new snapshot, it creates an envelope containing two packets and sends it through the P2P anonymous network. Inside the envelope, the former packet contains a transaction *id* and one of the image shards, generated according to our secret sharing scheme. The latter, instead, contains the same transaction *id*, the second image shadow, the timestamp, and a digest computed through a shared function. The digest function computes the digest of a secret message obtained by combining different information, including the timestamp, a secret seed shared between the camera and the *Home Server*, and a public nonce (see Section V-B for all details about it).

The envelope is a standard Onion message, whose payload contains both a content and a second Onion message. The smart camera generates a sequence of hops, peers in the P2P network, which must be involved in the delivery of the envelope. A part of these hops are used to build the internal Onion message; the others, instead, form the hops the whole envelope will perform in the anonymous P2P network. The

details about this procedure are reported in Section V-C. Of course, the last hops of both the envelope and the internal message coincide with the servers devoted to the data storage.

4) PHASE 4. SENDING THE ENVELOPE

The right sequence of hops to use in the delivery of the envelope (along with its internal message) is generated by the smart camera that acquires the new snapshot. This sequence is built by leveraging an ad-hoc random step function, which returns a variable number of smart camera identifiers extracted from the P2P network, at random. As will be clearer in Section V-C, the random step function leverages a secret seed, shared between the *Home Server* and the smart camera.

5) PHASE 5. RECONSTRUCTING THE ORIGINAL IMAGE

An important point to consider is that, to guarantee accountability in case of an accident, the message source along with the image and timestamp must be reconstructed. This is the only case in which, in our scenario, the *Home Server* and the *TTP Server* collude to restore the original information. In a real-life scenario, it is possible to assume that this collusion action is carried out as an answer to a public law enforcement agency in charge of conducting an investigation on a given accident. Having access to the information stored in both servers, it is possible to restore all the needed data. Indeed, as for the image, the application of the adopted secret sharing scheme guarantees a full recovery of the image if both the shadows are available. As said, the timestamp, instead, is stored in clear text in one of the servers.

Concerning the physical position in which the snapshot has been taken, this coincides with one of the corresponding smart camera and, hence, restoring the information about the source of the data about a specific snapshot stored in the servers allows for the retrieval of the position. For this purpose, the *Home Server* receives information about the timestamp and the digest computed by the origin smart camera. As said, all the smart cameras share a secret seed with the *Home Server*, which, hence, has a mapping between each smart camera and the corresponding secret. Therefore, to find the origin of a message, the *Home Server* adopts a *generate and test* approach leveraging the shared digest function with a suitable combination of the secret of each camera, the public nonce, and the timestamp, as input. The *generate and test* approach ends when the output digest matches the one generated by the origin of the message (which is stored by the *TTP server*).

In the next sections, we are going to describe the behavior of each actor of our model in detail.

B. HANDLING ACCESS EVENTS: THE SMART CAMERA PERSPECTIVE

The smart camera is the first component of our system activated by an access to one of the monitored areas. In our reference scenario, each entrance to a secure/monitored zone is equipped with a smart camera suitably located so that it can capture a snapshot of the face of any person entering the zone. Snapshots, along with the position of the camera (i.e., the specific entrance) and the timestamp, must be stored to being able to verify accesses in case of an accident.

The objective of each camera is to send to the monitoring servers anonymized information about the access of a person to a secure area. To do so, it adopts a combination of a secret sharing methods for images and an ad-hoc random step function to build a pseudo-random path across other cameras to conceal, but still preserving information about the person, the entrance position, and the access detection timestamp. Preliminarily, all the smart cameras and the *Home Server* share a public nonce released by the *Home Server*, a pseudo-random generator function, say $f_{PRNG}(s, n, k)$, and a cryptographic hash function, namely $f_{CHS}(message)$. In particular, $f_{PRNG}(s, n, k)$ generates n pseudo-random integer numbers in the interval $(0, k)$ and uses the value s as initial seed. Instead, $f_{CHS}(message)$ generates the digest of the input *message*. Moreover, each smart camera is equipped with a secret seed shared only with the *Home Server*.

With that said, the camera adopts the following steps. Whenever somebody enters the area surveilled by it, the camera captures a snapshot of the entering person's face. This snapshot, say sn , will have different information associated with it, such as the snapshot size, resolution and timestamp. As said, among the others, the information about access timestamps plays a crucial role in our approach.

Indeed, after this, the camera proceeds by applying a secret sharing methods for images to split the original snapshot into n shards. For this purpose, we adopt a modified version of

Shamir image sharing algorithm proposed by Thien and Lin and described in [20], which reduces the size of the shards to a fraction of the original image (see Section III for details about it). This is an interesting property in our application context because each packet is sent through the network of smart cameras before reaching the servers; therefore, maintaining the size of the packets small is an important requirement for improving performances. In our case, we set the number of layers $N = 3$ and the minimum number of shards $K = 2$. Observe that, in our scenario, we are considering the basic configuration assuming that we have just one *Home Server* and a *TTP server*. However, our approach is extendable and we could include a higher number of servers. In this case, we could increase the number of layers and the minimum number of shards to re-build the original image, so that we could assign one shard to each involved server. As for the other information, namely the timestamps, the smart camera seeds, and the computed digests, the only requirement imposed by our solution is to keep them separated across the servers owned by the organization and the trusted-third party servers.

After the construction of the image shards, the smart camera proceeds by computing the digest of the public nonce pn_{HS} provided by the *Home Server*. To do so, first it computes the message m as a concatenation of the public nonce pn_{HS} , the smart camera seed cs , and the timestamp of the snapshot ts_{sn} , as follows:

$$m = pn_{HS} || cs || ts_{sn}$$

Then, being l_m the length of the message, it computes a new message \hat{m} by shuffling the characters of m as follows:

$$\hat{m} = f_{SHF}(m, cs)$$

Here, given the original message m , the function $f_{SHF}(m, cs)$ shuffles the characters of m by applying the following steps. First, a list of size l_m of random numbers in the range $(0, l_m)$ is obtained by applying $f_{PRNG}(cs, l_m, l_m)$. Then, each character in \hat{m} is derived by picking the characters of m corresponding to the positions in the random list computed before.

Finally, the function $f_{CHS}(\hat{m})$ is applied to obtain the digest, say $d_{\hat{m}}$, of the shuffled message \hat{m} :

$$d_{\hat{m}} = f_{CHS}(\hat{m})$$

At this point, the smart camera proceeds by building the envelope to be sent across the P2P network of smart cameras (see Section V-C for all details about it). The envelope is an Onion package, which contains information about the hops that the envelop will perform in the P2P network and a payload. This last contains the transaction identifier, T_ID for short, one of the image shard and an additional Onion package. As stated before, the second Onion package will contain the transaction identifier, the second shard obtained from the original image, the timestamp ts_{sn} , and the digest $d_{\hat{m}}$. As a side note, we clarify that the transaction identifier

Algorithm 1 Assembling the Envelope

```

1: procedure GENERATING INITIAL INPUTS
2:   HS generates  $pn_{HS}$ ;
3:   for  $C_i$  in P2P do
4:     HS  $\rightarrow C_i : pn_{HS}$ ;
5:     HS  $\rightarrow C_i : f_{PRNG}(s, n, k)$ ;
6:     HS  $\rightarrow C_i : f_{CHS}(message)$ ;
7:   end for
8: end procedure
9: procedure GENERATING ENVELOPE
10:  if U enters a room  $i$  then
11:     $C_i$  takes  $sn$ 
12:     $C_i$  computes  $n = 3$  layers and  $Im_1, Im_2$  shards from  $sn$ 
13:     $C_i$  computes  $m = pn_{HS} || cs || ts_{sn}$   $\triangleright$  The message
14:     $C_i$  computes  $\hat{m} = f_{SHF}(m, cs)$   $\triangleright$  The shuffled message
15:     $C_i$  computes  $d_{\hat{m}} = f_{CHS}(\hat{m})$   $\triangleright$  The digest of the shuffled message
16:     $C_i$  computes  $path_1 : \langle C_1, C_3, HS \rangle$ 
17:     $C_i$  computes  $path_2 : \langle C_2, C_1, TTPS \rangle$ 
18:     $C_i$  computes  $T\_ID$   $\triangleright$  The transaction id
19:     $C_i$  creates:
20:     $env_{TTPS} = E_{TTPS}\{T\_ID, Im_2, ts_{sn}, d_{\hat{m}}\}$ 
21:     $env_{HS} = E_{HS}\{T\_ID, Im_1, C_2, E_{C_2}\{C_1, E_{C_1}\{TTPS, env_{TTPS}\}\}\}$ 
22:    envelope =  $\langle C_1, E_{C_1}\{C_3, E_{C_3}\{HS, env_{HS}\}\}\rangle$ 
23:  end if
24:  return envelope
25: end procedure

```

is needed to re-assemble the two messages in case of an enforce collusion action. Algorithm 1 shows all the steps for the preparation of the envelope before forwarding it in the P2P network.

C. AN ONION-BASED ROUTING TO CONCEAL ACCESS POINTS

Now, to conceal the source of the envelope, our approach leverages the Onion protocol and assumes that a random sequence of hops inside the P2P network is chosen in a pseudo-random way by the smart camera. In this way, the information about the last smart camera involved in the path towards each server cannot be used to derive the origin. In more detail, the smart camera has to generate a random path for the whole envelope and another one for the secondary Onion package that is part of the payload of the envelope.

For this reason, given the parameter $max_hop = max(2, f_{PRNG}(cs, 1, |C|))$ specifying the maximum number of hops for a package,¹ it generates two pseudo-random

¹Here, the max function computes the maximum value between 2, i.e. the minimum number of hops required by our system so that the information reaches the two servers, and a random value ranging in $(0, |C|)$.

sequences of hops using the function $f_{PRNG}(cs, \frac{max_hop}{2}, |C|)$, two times. Here, $|C|$ is the number of smart cameras in the environment.

After that, the smart camera will append the address of the *Home Server* to the former list and the address of the *TTP Server* to the latter. By using the two lists of hops it will build the internal Onion package and the whole envelope.

Figure 4 shows a representation of the obtained envelope.

In the example sketched in this figure, according to the Onion protocol, each layer of the envelope has the recipient address and a payload encrypted with the key of the recipient. The first recipient is the smart camera C_1 , which decrypts the message and obtains the address of the next recipient, i.e. the smart camera C_3 . This last decrypts its envelope layer and retrieves the address of the next recipient, which is the *Home Server*. The *Home Server* decrypts the remaining of the original envelope and retrieves the image shard, which it will store in its database, along with the corresponding transaction identifier and the address of the next recipient of the other part of the envelope, i.e., C_2 . At this point, the message will flow through the network following the same reasoning above and involving the nodes C_2, C_1 once again, and, finally, the *TTP Server*. This extracts the original transaction identifier and three pieces of information, namely the second image shard, the timestamp of the snapshot, and the digest, and proceeds to store them in its database.

D. HANDLING ACCESS EVENTS: THE SERVER PERSPECTIVE

As stated in the previous sections, our scenario assumes the presence of at least two servers, namely the *Home Server*, which is typically the server handled by the organization owner of the surveillance mechanism, and the *TTP Server*, which is a server provided by a trusted-third party external to the organization. Once again, it is worth noting that, although in this paper we are considering one *Home Server* and one *TTP Server*, our approach is scalable and it can work in a scenario in which more servers are available.

Anyway, in most of real-life scenarios, the basic configuration with two servers appears the most adequate. Moreover, it allows for a simpler description of our proposal, without loss of generality. Therefore, throughout this paper we make explicit reference to a two-servers scenario under the assumption of non-collusion, unless enforced by a law agency.

Under the previous assumption, both the servers participate to the protocol as special nodes of the P2P Onion-enabled network, because they can provide also permanent storage. In particular, they are in charge of storing the information received from the other nodes of the P2P network and/or to forward the remaining part of the message, if needed.

Whilst the role of the *TTP Server* is to simply store the data received and to provide access to them only if enforced by a law agency, the *Home Server*, i.e., the organization server, instead, is also involved in the information restoring process.

For this reason, it stores also information about the seeds of each smart camera (see Figure 5).

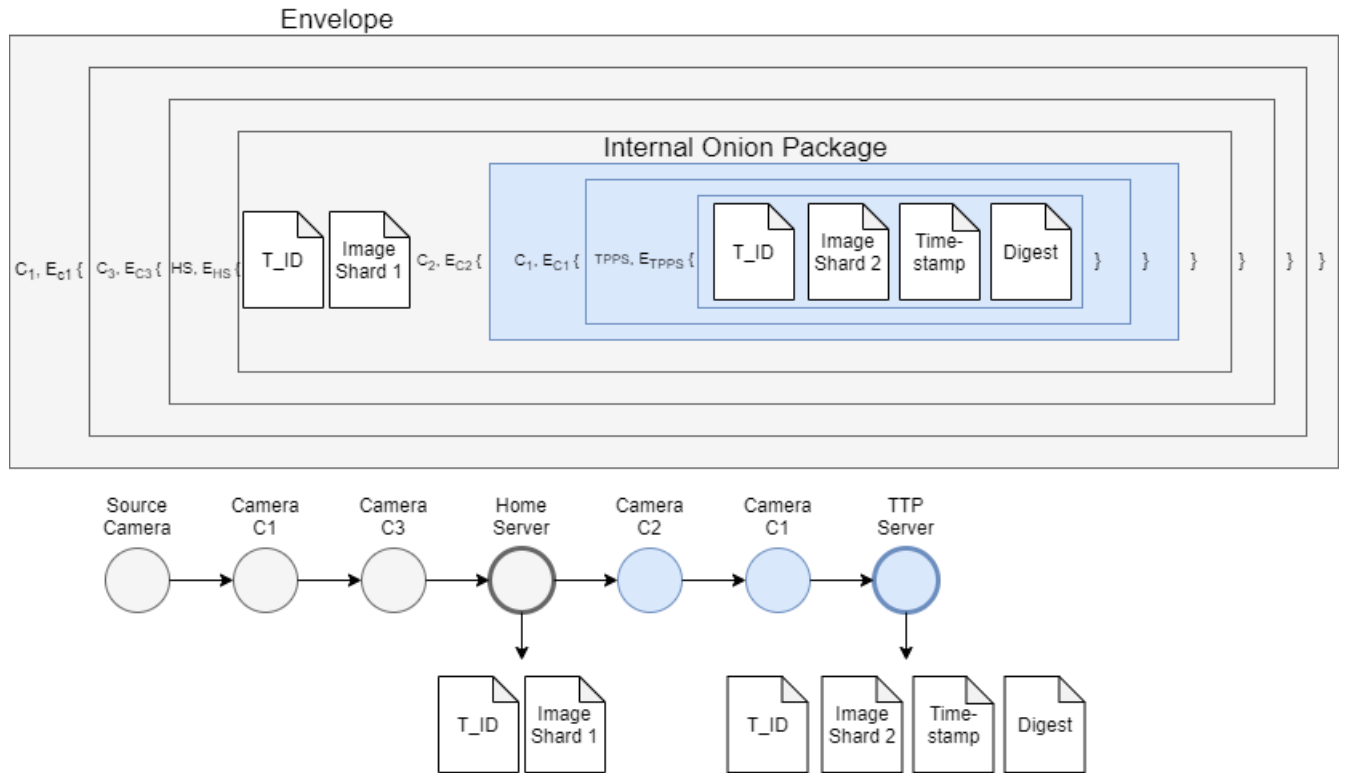




FIGURE 4. The envelope and the transmission strategy.



Home Server (HS)

T_ID	Image Shard
1234	1F8B080087CD...
1235	03F348CDC9C9...
...	



TTP Server (TTPS)

T_ID	Image Shard	Timestamp	Digest
1234	8CF2FCA49D1E...	2021-01-01 09:00:01	900150983cd...
3246	30200D7BBV4...	2021-01-01 09:03:21	6963f7d28e...
...			

Smart camera ID	Seed
1	37098
2	56845
...	

FIGURE 5. Tables related to the Home Server and the TTP Server.

In the next section, we will describe the mechanism adopted to restore the original information to enable accountability in case of an accident.

E. RESTORING INFORMATION FOR ACCOUNTABILITY

In the previous sections, we described our strategy to provide an anonymous surveillance mechanism. However, one of the fundamental claims in our proposal is that our approach provides full accountability capabilities in case of a critical event. Indeed, as stated above, our solution

leverages at least two servers, one owned by the organization, and one external handled by a trusted-third party, to split information about access events and store them separately to inhibit the retrieval of the original data on either of the two parties. However, our approach provides an information restoring capability leveraging the on-demand collusion of the two servers. Once again, in real life scenarios, the on-demand collusion can be triggered by any law enforcement agency in case of an accident or a critical event.

Of course, to minimize the information revealed, the agency could lead the collusion action by querying the trusted-third party and returning only the essential information to the organization. Indeed, it could be possible to retrieve information of the events which have been recorded in a specific time interval² and, then, proceed by retrieving the digests corresponding to each record in the time interval for filtering out the events recorded at the entrance of specific areas (the details about how to restore information about the specific entrance starting from a digest will be explained below). Finally, the agency could retrieve the shards corresponding to the events of interest from the *TTP Server*.

With that said, the restoring mechanism works as follows. First of all, given the transaction identifier and the digest provided by the *TTP Server*, the *Home Server* derives the identifier of the smart camera that captured the snapshot. To do so, it starts from the set $M_{cs} = \{c_i, cs_i\} \forall c_i \in C$ containing the mapping between each smart camera c_i and the corresponding internal seed cs_i . At this point, for each element of M_{cs} , it builds a message $m_i = pn_{hs} || cs_i || ts_{sn}$ by concatenating the public nonce pn_{hs} , the camera seed c_i , and the snapshot timestamp ts_{sn} provided by the *TTP Server*. By doing so, it obtains the set $M'_{cs} = \{c_i, cs_i, m_i\} \forall c_i \in C$.

After that, for each element of M'_{cs} , it applies the shuffling function $f_{SHF}(m_i, cs_i)$ to the message m_i by using the seed cs_i , thus obtaining the new set $\widehat{M}'_{cs} = \{c_i, cs_i, \widehat{m}_i\} \forall c_i \in C$. Hence, once again, for each element of \widehat{M}'_{cs} , it computes a digest for the shuffled message \widehat{m}_i by applying the cryptographic hash function $f_{CHS}(\widehat{m}_i)$; in this way it obtains the final set $M^d_{cs} = \{c_i, cs_i, d_{\widehat{m}_i}\} \forall c_i \in C$. Now, the *Home Server* compares all the digests in the tuples of M^d_{cs} with the one returned by the *TTP Server* for this specific transaction identifying the matching one. In this way, the *Home Server* is able to obtain the identifier c_i of the smart camera that captured the snapshot and, hence, the position, i.e., the entrance it is surveying.

Finally, it is possible to proceed with the retrieval of the image shard of the *TTP Server* for this transaction and to reconstruct the original snapshot by inverting the image secret sharing algorithm and adopting the restore mechanism described in [20].

VI. SECURITY MODEL

In this section, we describe the security model. In particular, we present both the attack model (Section VI-A) and a security analysis (Section VI-B) showing that our system is robust also in presence of attacks. We will refer to both classical attacks typical of Onion-based scenarios and direct attacks to the involved entities.

A. ATTACK MODEL

Preliminarily, to ensure the correct implementation of our solution, we assume that a sufficient number of cameras is available in our system.

²Recall that the *TTP Server* stores information about the timestamp of each record.

In addition, our approach focuses on the protection against privacy leakages deriving from the availability of access log information from a video surveillance based access monitoring system, therefore we do not consider direct threats to the P2P communication channel, such as DoS attacks [60]. Anyway, these menaces are thoroughly studied in the scientific literature and, therefore, even if the identification of suitable strategies for the protection from them is orthogonal to our solution, it is possible to leverage any available solution already proposed in the literature to make our approach robust also against these kinds of attack [61]–[63].

Finally, in the analysis of security properties, we will assume that our threat model includes the following statements:

- **A.1** - An attacker cannot control multiple smart cameras in our environment.
- **A.2** - An attacker cannot control nor monitor the whole P2P network.
- **A.3** - An attacker cannot have access to both the *Home Server* and the *TTP Server*.
- **A.4** - The *Home Server* and the *TTP Server* do not collude without an explicit request from a law enforcement agency.
- **A.5** - The adopted image secret sharing algorithm is robust and cannot be inverted without the expected number of image shards.
- **A.6** - The adopted cryptographic hash function is assumed secure and non-invertible.
- **A.7** - The adopted PRNG function is cryptographically secure (CS-PRNG).
- **A.8** - An attacker has no additional background knowledge derived from the environment or any direct physical access to the system entities.

Given the assumptions above, we can identify the security properties of our solution. Specifically:

- **SP.1** - Resistance to timing attacks to reconstruct packet routes.
- **SP.2** - Resistance to package correlation attacks.
- **SP.3** - Resistance to attacks to the P2P network to force controlled paths.
- **SP.4** - Resistance to behavior-based deanonymization or traffic classification attacks.

B. SECURITY ANALYSIS

In this section, we study each security property in details and discuss about the capability of our approach to guarantee it.

1) **SP.1** - RESISTANCE TO TIMING ATTACKS TO RECONSTRUCT PACKET ROUTES

This property guarantees that an attacker cannot obtain information about the route of a packet (e.g., an envelope built according to the strategy described in Section V-B) by performing a timing attack on the nodes of the Onion-based P2P network adopted in our system. In our scenario, this typology of attacks would lead to a potential vulnerability only if the

attacker should have also access to the *TTP Server*. Indeed, the strategy underlying it relies on the possibility for the attacker to analyze the time taken by each node in the network to forward a packet. In practice, if an attacker can observe the packets in input and output of a node, he can try to map each input to an output by estimating an elaboration time for each packet.

In the trivial case in which all the packets have the same structure and require the same elaboration tasks, intuitively, the attacker could assume a *FIFO* policy on the packet forwarding (see Sections III and V-C), and, therefore, he can easily perform the needed mapping. This can, then, be used to invert the path and obtain a link between the data stored in one of the server and the data source. In particular, if the attacker should also gain access to the *TTP Server*, this attack would give the adversary a link between the source of the packet and its content. Knowing the source, the attacker could exploit the content of the packet, i.e., the digest and the timestamp, to try a brute force attack and obtain the seed of the source smart camera. Indeed, the attacker knows that the seed is used together with a public nonce from the *Home Server* and the snapshot timestamp to build a basic message to compute the digest. However, our approach leverages a strong protocol to compute this digest in such a way as to avoid providing any advantage to an adversary. In fact, starting from the basic message above, our approach performs a shuffling operation on it, before proceeding with the computation of the digest. The shuffling operation is performed by leveraging a strong cryptographically secure PRNG function and by using, once again, the seed of the camera as initial value. Now, the attacker knows only the digest of the shuffled message and, therefore, he has no advantages to inverting the cryptographic hash function adopted to compute the digest. For this reasons, and thanks to Assumptions A.3, A.6, and A.7, the attacker cannot revert the digest and obtain the seed of the source smart camera. Furthermore, thanks to Assumption A.5, the attacker cannot exploit the knowledge of a piece of the image to reconstruct the original snapshot and link it to a specific entrance (i.e., to the position of the source smart camera).

However, this attack can still break our privacy setting because the attacker could link the access-event timestamp to a specific location in the surveyed environment. Of course, this is a minor issue and, thanks to Assumption A.8, it cannot lead to more severe privacy leakage. Still, a simple strategy could be adopted to prevent even this minor vulnerability. Indeed, the basic assumption under this timing attack is that each packet has the same dimension and elaboration time. Therefore, in the absence of any particular routing protocol, the attacker may assume that each node adopts a *FIFO* policy to forward received packets. To prevent the attacker from having any advantage from this assumption, in our approach we could force each node of the Onion-based P2P network to introduce random delays to input packets before forwarding them. In this way, the *FIFO* assumption does not hold anymore, thus avoiding that the attacker can perform a timing

attack and, hence, follow a packet from the source smart camera to the *TTP Server*.

2) **SP.2** - RESISTANCE TO PACKAGE CORRELATION ATTACKS

The aim of this property is protecting our system from attacks based on the possibility, for the attacker, to obtain advantages by correlating different transactions available in the *TTP Server*. As a prerequisite, also in this case, he must have access to the *TTP Server*. In this case, the attacker tries to find patterns in the transactions (i.e., data related to access events) stored in the server with the objective of finding clusters of access events coming from the same smart cameras. A variant of this attack could include a data injection task, in which the attacker may organize a simultaneous physical access to a specific entrance of a group of people and, hence, identify the corresponding transactions on the *TTP Server* by observing the associated timestamps.

In both cases, the attacker will have a cluster of transactions corresponding to a specific smart camera. With this information at disposal, he will try, once again, to invert the digest and retrieve the smart camera seed.

Also in this case, this attack cannot happen. As a first observation, thanks to Assumptions A.6, the attacker cannot invert the hash to obtain the original message. However, in this case, he has the advantage of knowing part of the original message (i.e., the timestamp and the public nonce). This could reduce the search space for a brute force attack to the hash function. To prevent even this advantage, our approach executes a shuffling operation to the original message (see Section V-B). As already stated in Section VI-B1 and thanks to Assumption A.7, this operation leverages a cryptographically secure PRNG. The fundamental properties of such a PRNG are its resistance to both the *next-bit test* [64] and the *state compromise extension* [65]. These properties prevent the attacker from being able to acquire any advantage in breaking the shuffling operation and obtaining a clear connection between the original message (containing the smart camera seed) and the digest.

3) **SP.3** - RESISTANCE TO ATTACKS TO THE P2P NETWORK TO FORCE CONTROLLED PATHS

In scenarios leveraging P2P networks, a common typology of attacks relies on the possibility of compromising the network itself and to force the delivery of packets to specific intermediary nodes. These nodes are typically under the attacker control or can be constantly monitored and, therefore, represent an oracle for the attacker that can, hence, exploit the obtained information. One of the main method to perform this attack in a P2P network is the application of selective Denial of Service attacks (DoS, for short) to specific nodes to favor the traffic towards specific paths [66].

In our scenario, the attacker may try to force the traffic towards specific sub-sets of the nodes so that he can identify the source of each packet systematically. This would cause a flaw in our privacy-preserving access monitoring strategy.

However, in our approach packet routing among nodes is established by the source smart camera randomly and cannot be changed (also due to the application of the Onion-based strategy) once the packet has left the source camera. Therefore, even a selective DoS attack would result in the impossibility of reaching the destination and, eventually, in a full DoS for our monitoring service. As stated above, in this paper we do not consider direct threat to the whole P2P communication channel and, therefore, we do not deal with this aspect. However, as stated above, the recent scientific literature report several solutions to DoS attacks [61]–[63]. Any of them could be integrated in our approach, thus making it robust also to full DoS attacks.

4) **SP.4** - RESISTANCE TO BEHAVIOR-BASED DEANONYMIZATION OR TRAFFIC CLASSIFICATION ATTACKS

This property deals with attacks based on the possibility of assuming peculiar characteristics of the traffic as well as identifying access patterns in specific entrances. In particular, in the former case the attacker tries to derive unique features in the traffic generated by each node involved in the P2P network. As an example, consider the case in which some devices should record videos (or capture snapshots) in full high-definition resolution, whereas others should work in standard definition mode. With this knowledge at disposal, the attacker can detect the operation mode of each smart camera by monitoring the size of transmitted packets. In this way, he can build a map of the network with the information about the typology of traffic generated and, hence, he could use it to relate specific transactions in the servers to a possible (even small) set of smart cameras (and, hence, physical entrances). However, in our scenario each smart camera is an exact replica of the others when it comes to their behavior. Therefore, there is no noticeable difference in the packets generated by them. This aspect, together with Assumptions A.2 and A.8, makes our approach robust against this types of attack.

Concerning the latter case above, in this situation the attacker would try to identify unique access patterns for different entrances. The idea, once again, is to link the transactions on the servers to specific locations (i.e., source smart cameras) in which the corresponding entrances have been recorded with the highest probability. However, this type of attacks requires that the attacker has specific information about the building so that he can hypothesize and measure specific patterns, such as the presence of a main entrance or specific movement habits of the users during each day. However, this kind of information requires a deep background knowledge of the environment and, in most cases, an authorized access to it. As discussed in [67], background information can always cause privacy leakages. Indeed, this concerns situations in which an attacker has already enough information and does not need to find vulnerabilities to bypass any protection mechanism. For this reason, the assumption of no background knowledge is commonly adopted in scientific literature [22], [68]–[70]. Therefore, in our application context,

through Assumption A.8, we assume that the attacker has no background knowledge about the monitored building (e.g., number of users in a building or area) or the users' movement habits inside it. As a consequence, also this type of attacks cannot happen in our case.

VII. EXPERIMENTS

In this section we describe the experiments we performed to test the feasibility of our approach and to determine the best configurations to satisfy the needed privacy requirement.

To carry out our experimental evaluation of the performance and features of our approach, we built a simulator based on the physical characteristics of a real-life environment to be monitored. We developed this prototype using Java language (version 8) on an Ubuntu Server equipped with 32 GB of RAM and a 8 core CPU at 5.10 GHz. In our case, the considered physical environment has an overall dimension of 500 squared meters (10×50 meters). Starting from this information, our simulator considers four different configurations of the physical building, each characterized by a different number of areas (also referred as rooms, in the following) and, hence, entrances, ranging from 8 to 64.

Also, the number of users, randomly moving inside the available areas, varies from 10 to 100. To simulate people movement inside, we adopted a custom version of a common mobility model typical of these scenarios, namely the Random Waypoint Mobility Model [71]. It generates shifts according to the following strategy. First, for each person inside the building, it selects a destination point in an area of the environment in input. After that, it simulates each person's movements towards the destination area for a walking interval (w_{min}, w_{max}) with a step speed selected uniformly at random in the interval (s_{min}, s_{max}). In our scenario, we assume that the speed and the direction of each person inside the environment are independent from those of the others. After reaching a destination or ending a walking interval, each person stands in the current spot for a time interval driven by the parameter *pause*, randomly selected in the range (p_{min}, p_{max}). After the pause interval, he continues his path towards the destination or selects another destination and starts moving towards it. In our simulator, we set $w_{min} = 2$ seconds, $w_{max} = 6$ seconds, $s_{min} = 0.2$ m/s, $s_{max} = 2.2$ m/s, $p_{min} = 0$ seconds, and $p_{max} = 1$ seconds.

We started our experimental campaign by comparing our approach with a naive one in terms of the load generated on the P2P network to reach the servers. In the naive approach, each smart camera sends data to the two servers directly, without applying any strategy to conceal the source. Hence, this approach consists of the following steps:

- At the entrance of each monitored zone, the smart cameras capture snapshots of users accessing them.
- Each image is split into three shards; only two shards are needed to reconstruct the original image (see Section III-A for details).
- The smart camera sends the first shard to the *Home Server*.

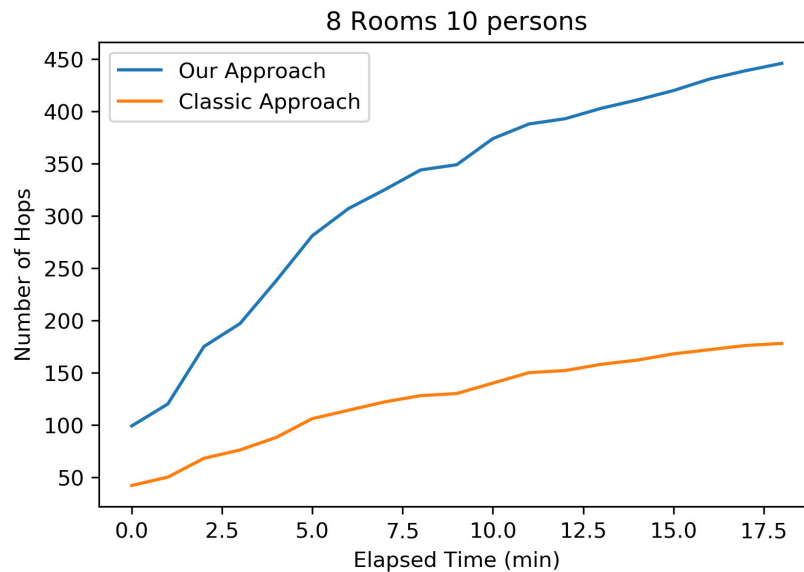


FIGURE 6. Comparison of our solution with a naive approach.

- The smart camera sends the second shard to the *TTP Server*.

As for the network performance, we measured the generated load in terms of the number of hops (number of involved smart cameras in the P2P network) required to deliver each packet to the destinations. In this first experiment, we considered a basic configuration of the monitored environment with 8 smart cameras and 10 persons moving inside of it for 1,050 seconds (17.5 minutes) according to our mobility model. In particular, we measured the number of packet exchanges (i.e., number of network hops) occurred during the observed time period. The results of this experiment are reported in Figure 6.

By observing this figure, we can have a better understanding of the impact of the protection mechanism introduced by our solution to the traffic generated in the network. Indeed, while the naive strategy generates a number of hops ranging from 50 to 180 during the whole time period, our approach introduces an overhead mostly related to the Onion-based routing and the presence of two servers (i.e., the *Home Server* and the *TTP Server*). In our case, the number of hops in the whole period ranges from 100 to 450, which basically doubles the traffic in the network, on average. However, it is worth noting that, in our application scenario, the P2P network is dedicated to the communications between smart cameras and servers and, therefore, the major impact in the traffic brought about by the security strategy does not compromise other services. Moreover, our solution is designed for a-posteriori analyses of access logs in case of accident and, therefore, the possible latency introduced by traffic overheads does not impact the overall system usability.

With that said, it is clear that the overhead in terms of hops generated before reaching the destinations is strictly related to the number of entrances/rooms and, hence, smart cameras

involved in the system. To test this aspect, we increased the number of smart cameras in the monitored environment and repeated the experiment above. In particular, we considered four configurations each characterized by a different number of rooms, equipped with a smart camera, in the environment. Specifically, we considered 8, 16, 32, and 64 rooms. In each configuration, we maintained the number of persons and their walking model unchanged. Once again, we simulated the movements, inside the building, of 10 people for 1,050 seconds and executed our protocol every time a person trespassed the area monitored by a smart camera. Once again, in Figure 7, we report the traffic generated by our approach in the P2P network in terms of cumulative number of hops.

The results of this analysis can be used as a reference to properly dimension the P2P network. Indeed, as the number of monitored areas in the environment grows, the traffic generated increases leading to a maximum of 2,500 package exchanges in a 1,050 seconds time window for a configuration with 64 monitored rooms.

Of course, the performance of our approach also depends on the number of people involved in the environment. Also in this case, the higher this number, the higher the probability of registering an entrance in one of the monitored areas.

To test the impact of a growing number of people in the monitored environment on our approach, we carried out a further experiment. In this case, we kept fixed the number of monitored rooms (and, hence, of the smart cameras) to 8 in the environment, and considered five different configurations, each characterized by an increasing number of people moving in the reference environment. Specifically, we considered 20, 40, 60, 80, and 100 persons. For each configuration, we simulated people's movements for about 20 minutes and reported the cumulative traffic generated in the P2P network (once again, in terms of number of hops)

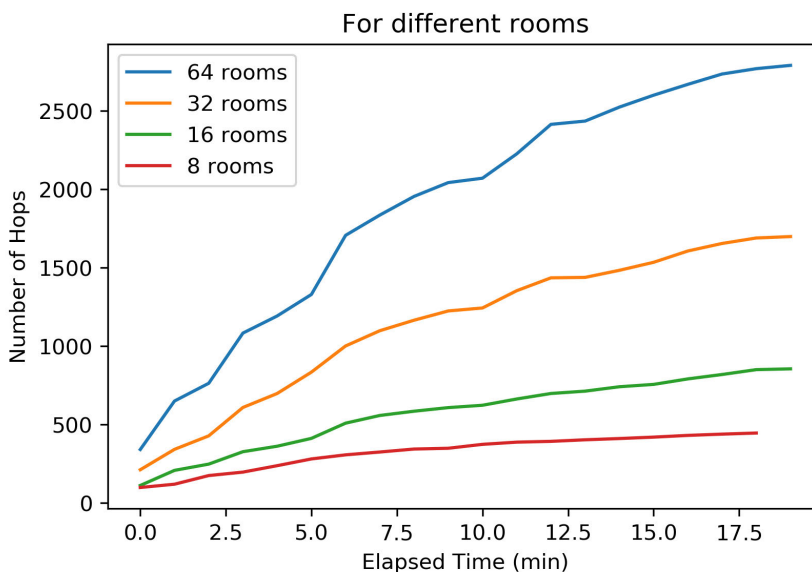


FIGURE 7. Comparison of different number of rooms.

every 5 minutes. Figure 8 shows the variation of the traffic generated in the network with respect to the number of users and the time elapsed.

The result of this experiment shows that the increase in the traffic generated by the smart cameras is linear over time as the number of persons increases.

Up to this point, we have tested the impact of different configurations, in terms of number of monitored areas and persons, to the traffic generated by our approach in the P2P network. Even if, as already stated above, in our referring context, the P2P network is fully dedicated and not shared with other services, and our solution is designed for a-posteriori (offline) analyses of access logs, it is interesting to evaluate the latency introduced by our solution for the delivery of packets to the servers. This will give an important insight to understand after how long access logs will be available for a-posteriori analyses in case of accidents. For this reason, we consider that each smart camera operates in full HD mode, thus capturing snapshots with a $1,920 \times 1,080$ resolution. Moreover, using a Python script leveraging OpenCV, each smart camera extracts automatically the area containing the face of the person from the image. The final snapshot has a size of 480 KB, on average. Under this configuration, we tested the execution of the image secret sharing algorithm and we measured an average elaboration time necessary to decompose the original snapshot into the required number of shards, equal to 1.8 seconds. Moreover, in our context we are making explicit reference to a scenario in which two servers (i.e., the *Home Server* and the *TTP Server*) are present. Therefore, we set the number of shards to 3 and our approach is able to obtain the original image using 2 of them (see Section III for all the details about the image secret sharing algorithm). In this case, the average size of a shard is equal to 160 KB. To compute the overall time required

for each packet to reach the destinations, we averaged the delivery times recorded in a scenario with 60 people and 32 monitored rooms (i.e., 32 smart cameras). In particular, we obtained that a single hop between two cameras of the whole envelope (containing both the shards) requires 960 ms, on average. We also assume that the inner Onion package, containing only one of the shards, requires about half of the time (480 ms, on average). Moreover, the envelope reaches the *Home Server* in 6,720 ms, and the second part of the packet reaches the *TTP Server* in 3,360 ms, on average.

In conclusion, the access log can be available for the analysis in about 10 seconds after the snapshot acquisition. Of course, this delay depends on the number of people and monitored rooms in the environment; lower values can be obtained in scenarios with less involved people or monitored areas. The results reported in this experiment should give an insight of the order of magnitude of the time needed to obtain the anonymous access logs. Anyway, this result appears perfectly in line with the objective of our approach, which is not to provide a real-time tool for access monitoring, but rather a system for privacy preserving access logs based on video surveillance.

Further reasoning about bandwidth and energy consumption can be made, although these metrics are strictly related to the particular hardware and communication protocol employed. The choice of the smart camera is orthogonal to our system. Anyway, typically, the energy consumption of low-bandwidth smart cameras spans from several hundreds of milliwatts (e.g., 428 – 478 mW) to some watts while performing intensive operations also in relation to the processor speed (e.g., 970 – 1500 mW).

As for the bandwidth consumption, once again, it strongly depends on the chosen image resolution and the adopted

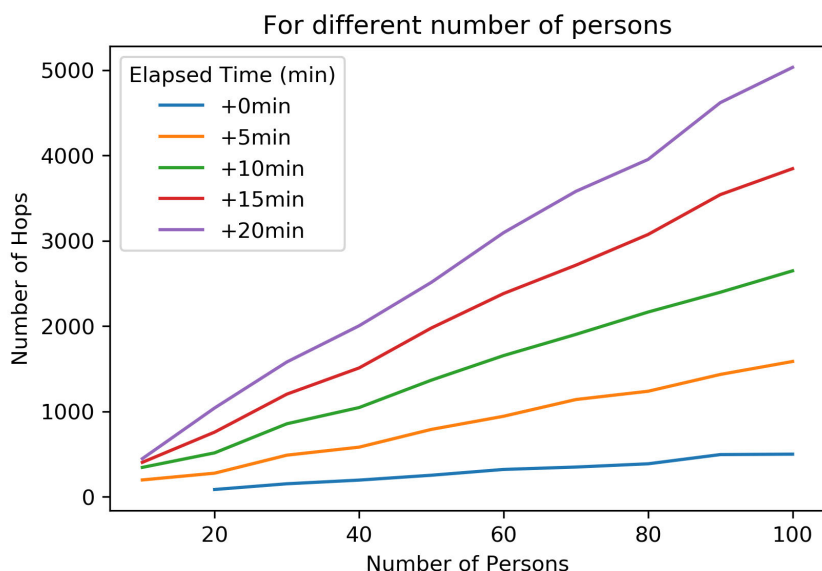


FIGURE 8. Comparison of different number of persons.

communication protocol. Anyway, for security cameras, typical bandwidth consumption can range from 5 Kbps to 6 Mbps.

VIII. DISCUSSION

In this section, we discuss the limitations of our proposal when applied in real-life scenarios. We recall that our approach has been specifically designed as a solution to generate anonymous access monitoring logs, in which information about the physiognomy of subjects, although anonymized, can be derived in case of accident. Our solution imposes that logs can be de-anonymized only in presence of a law enforcement agency. Such a task is performed by means of the controlled collusion between the service owner and the trusted third party.

However, by design, our strategy cannot be adopted in real-time or pseudo-real-time applications in which access data must be monitored constantly. Although the objective of our proposal is not real-time monitoring, still it can be combined with existing solutions, such as the approach described in [72], allowing for the real-time detection of specific access events, to include such a feature when and if needed.

Another important point to consider is that our approach has been developed by taking new generation smart-cameras in mind. Perfectly in line with the prerequisites of IoMT scenarios, it assumes that the devices adopted for image (video frame) acquisitions have higher computational capabilities than the most common objects of a standard IoT context. This can represent a limitation of our approach. However, in a scenario in which not all the cameras have sufficient computation power, our approach can be adapted so that not all the nodes in the P2P network are responsible for the application of the image secret sharing algorithm (which is the most computational demanding part of our solution). In this case, we may distinguish between ordinary smart cameras

and full-equipped ones. The latter cameras can be leveraged by ordinary ones to delegate all or part of their computations. However, such an extension in our scheme may come with a reduction of the overall security properties of our solution.

As a final observation, our approach adopts a robust image secret sharing algorithm to protect acquired images. However, in many application scenarios, video-surveillance can be adopted to enable access control. Our scheme has been built to provide privacy preserving access monitoring and it does not support facilities for access control. This limitation is also a consequence of not considering real-time monitoring in our proposal. Therefore, once again, in scenarios in which real-time elaboration of at least some part of the access information is required, our approach could be combined with existing ones to include this functionality.

IX. CONCLUSION

In this paper, we have analysed and described a distributed solution, in the context of IoMT, for private-preserving access monitoring based on video-surveillance. One of the main novelties of the proposed solution is that it protects against server-side attacks to privacy also in the case in which the system owner cannot be considered trusted. The different information held by the actors of the system is not sufficient alone to re-identify the subject or to track him. Moreover, our system retains the ability to reconstruct the original data, thus providing full accountability in case of a legal requirement. Our proposal makes use of a network of smart cameras located at the entrance of monitored zones to capture people's face snapshots. After the application of an ad-hoc image secret sharing algorithm, smart cameras send the shards to at least two separated and non-colluding servers using a P2P network powered with an Onion-based routing protocol. Through an extensive experimental campaign, we evaluated

the performance of our approach for a different number of monitored areas and people in a reference test environment. The research described in this paper must not be considered as an ending point. Indeed, we are already studying future extensions of it. For instance, we plan to design a system leveraging both this approach and Machine Learning techniques, useful to anonymously categorize people in some defined classes (e.g., professors or students in a University). Moreover, we plan to build a detailed architecture capable of deriving the network of physical contacts among people still preserving their privacy. Such an approach could be adopted in several application contexts, such as those related to the monitoring of the spread of an infection, which is a topic very actual and relevant at the time of writing this paper.

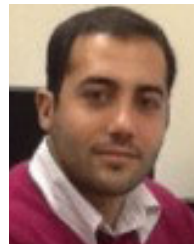
REFERENCES

- [1] E. L. Piza, B. C. Welsh, D. P. Farrington, and A. L. Thomas, "CCTV surveillance for crime prevention: A 40-year systematic review with meta-analysis," *Criminol. Public Policy*, vol. 18, no. 1, pp. 135–159, Feb. 2019.
- [2] M. P. J. Ashby, "The value of CCTV surveillance cameras as an investigative tool: An empirical analysis," *Eur. J. Criminal Policy Res.*, vol. 23, no. 3, pp. 441–459, Sep. 2017.
- [3] Y. Ma, Y. Wu, J. Ge, and J. Li, "An architecture for accountable anonymous access in the Internet-of-Things network," *IEEE Access*, vol. 6, pp. 14451–14461, 2018.
- [4] T.-H. Tan, M. Gochoo, F.-R. Jean, S.-C. Huang, and S.-Y. Kuo, "Front-door event classification algorithm for elderly people living alone in smart house using wireless binary sensors," *IEEE Access*, vol. 5, pp. 10734–10743, 2017.
- [5] L. Menard, "Is public video surveillance a right of government?" in *Proc. E-Learn, World Conf. E-Learn. Corporate, Government, Healthcare, Higher Educ.*, New Orleans, LA, USA, San Diego, CA, USA: Association for the Advancement of Computing in Education, 2019, pp. 949–954.
- [6] Y. E. Appenzeller, P. S. Appelbaum, and M. Trachsel, "Ethical and practical issues in video surveillance of psychiatric units," *Psychiatric Services*, vol. 71, no. 5, pp. 480–486, May 2020.
- [7] S. Nicolazzo, A. Nocera, D. Ursino, and L. Virgili, "A privacy-preserving approach to prevent feature disclosure in an IoT scenario," *Future Gener. Comput. Syst.*, vol. 105, pp. 502–519, Apr. 2020.
- [8] Y. B. Zikria, M. K. Afzal, and S. W. Kim, "Internet of multimedia things (IoMT): Opportunities, challenges and solutions," *Sensors*, vol. 20, no. 8, 2020, Art. no. 2334.
- [9] S. A. Alvi, B. Afzal, G. A. Shah, L. Atzori, and W. Mahmood, "Internet of multimedia things: Vision and challenges," *Ad Hoc Netw.*, vol. 33, pp. 87–111, Oct. 2015.
- [10] A. Nauman, Y. A. Qadri, M. Amjad, Y. B. Zikria, M. K. Afzal, and S. W. Kim, "Multimedia Internet of Things: A comprehensive survey," *IEEE Access*, vol. 8, pp. 8202–8250, 2020.
- [11] A. Schwartz, "Chicago's video surveillance cameras: A pervasive and poorly regulated threat to our privacy," *Nw. J. Tech. Intell. Prop.*, vol. 11, no. 2, p. 9, 2012.
- [12] H. Kim, Y. Cha, T. Kim, and P. Kim, "A study on the security threats and privacy policy of intelligent video surveillance system considering 5G network architecture," in *Proc. Int. Conf. Electron., Inf., Commun. (ICEIC)*, Jan. 2020, pp. 1–4.
- [13] A. Prati, R. Vezzani, M. Fornaciari, and R. Cucchiara, "Intelligent video surveillance as a service," in *Intelligent Multimedia Surveillance*. Berlin, Germany: Springer-Verlag, 2013, pp. 1–16.
- [14] J. Atick, P. Griffin, and A. Redlich, "Continuous video monitoring using face recognition for access control," U.S. Patent 6 111 517, Aug. 29, 2000.
- [15] H. Lee, S.-H. Park, J.-H. Yoo, S.-H. Jung, and J.-H. Huh, "Face recognition at a distance for a stand-alone access control system," *Sensors*, vol. 20, no. 3, p. 785, Jan. 2020.
- [16] L. Fu and X. Shao, "Research and implementation of face detection, tracking and recognition based on video," in *Proc. Int. Conf. Intell. Transp., Big Data Smart City (ICITBS)*, Jan. 2020, pp. 914–917.
- [17] T. Velavan and C. Meyer, "The COVID-19 epidemic," *Tropical Med. Int. Health*, vol. 25, no. 3, p. 278, 2020.
- [18] E. Bentafat, M. Rathore, and S. Bakiras, "A practical system for privacy-preserving video surveillance," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.*, Rome, Italy. Cham, Switzerland: Springer, 2020, pp. 21–39.
- [19] A. Bouti and J. Keller, "Towards practical homomorphic encryption in cloud computing," in *Proc. IEEE 4th Symp. Netw. Cloud Comput. Appl. (NCCA)*, Jun. 2015, pp. 67–74.
- [20] C.-C. Thien and J.-C. Lin, "Secret image sharing," *Comput. Graph.*, vol. 26, no. 5, pp. 765–770, Oct. 2002.
- [21] A. H. H. Basri, S. N. Ibrahim, N. A. Malik, and A. L. Asnawi, "Integrated surveillance system with mobile application," in *Proc. 7th Int. Conf. Comput. Commun. Eng. (ICCCE)*, Sep. 2018, pp. 218–222.
- [22] F. Buccafurri, G. Lax, S. Nicolazzo, and A. Nocera, "A privacy-preserving localization service for assisted living facilities," *IEEE Trans. Services Comput.*, vol. 13, no. 1, pp. 16–29, Jan. 2020.
- [23] F. Buccafurri, G. Lax, S. Nicolazzo, and A. Nocera, "A privacy-preserving solution for tracking people in critical environments," in *Proc. IEEE 38th Int. Comput. Softw. Appl. Conf. Workshops*, Jul. 2014, pp. 146–151.
- [24] B. QasemiZadeh, J. Shen, I. O'Neill, P. Miller, P. Hanna, D. Stewart, and H. Wang, "A speech based approach to surveillance video retrieval," in *Proc. 6th IEEE Int. Conf. Adv. Video Signal Based Surveill.*, Sep. 2009, pp. 336–339.
- [25] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in Internet-of-Things," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1250–1258, Oct. 2017.
- [26] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y.-L. Tian, A. Ekin, J. Connell, C. Fe Shu, and M. Lu, "Enabling video privacy through computer vision," *IEEE Secur. Privacy Mag.*, vol. 3, no. 3, pp. 50–57, May 2005.
- [27] F. Dufaux and T. Ebrahimi, "Scrambling for privacy protection in video surveillance systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 8, pp. 1168–1174, Aug. 2008.
- [28] E. M. Newton, L. Sweeney, and B. Malin, "Preserving privacy by de-identifying face images," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 2, pp. 232–243, Feb. 2005.
- [29] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–7.
- [30] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. 1st Int. Conf. Mobile Syst., Appl. Services (MobiSys)*, 2003, pp. 31–42.
- [31] S. Tansuriyavong and S.-I. Hanaki, "Privacy protection by concealing persons in circumstantial video image," in *Proc. Workshop Perceptive User Interfaces (PUI)*, 2001, pp. 1–4.
- [32] T. Koshimizu, T. Toriyama, and N. Babaguchi, "Factors on the sense of privacy in video surveillance," in *Proc. 3rd ACM Workshop Continuous Archival Retrieval Pers. Experiences (CARPE)*, 2006, pp. 35–44.
- [33] F. Z. Qureshi, "Object-video streams for preserving privacy in video surveillance," in *Proc. 6th IEEE Int. Conf. Adv. Video Signal Based Surveill.*, Sep. 2009, pp. 442–447.
- [34] J. Wickramasuriya, M. Datt, S. Mehrotra, and N. Venkatasubramanian, "Privacy protecting data collection in media spaces," in *Proc. 12th Annu. ACM Int. Conf. Multimedia*, 2004, pp. 48–55.
- [35] M. Saini, P. K. Atrey, S. Mehrotra, S. Emmanuel, and M. Kankan, "Privacy modeling for video data publication," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2010, pp. 60–65.
- [36] J. W. Kim, D.-H. Kim, and B. Jang, "Application of local differential privacy to collection of indoor positioning data," *IEEE Access*, vol. 6, pp. 4276–4286, 2018.
- [37] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. V. Jawahar, "Efficient privacy preserving video surveillance," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 1639–1646.
- [38] C. Gentry, *A Fully Homomorphic Encryption Scheme*, vols. 20–29. Stanford, CA, USA: Stanford Univ., 2009.
- [39] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for Web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, no. 1, pp. 66–92, Nov. 1998.
- [40] P. Syverson, D. Goldschlag, and M. Reed, "Onion routing for anonymous and private Internet connections," *Commun. ACM*, vol. 42, no. 2, p. 5, 1999.
- [41] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Commun. ACM*, vol. 42, no. 2, pp. 39–41, 1999.
- [42] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervas. Comput.*, vol. 2, no. 1, pp. 46–55, Jan. 2003.
- [43] R. Want, A. Hopper, V. Falcão, and J. Gibbons, "The active badge location system," *ACM Trans. Inf. Syst.*, vol. 10, no. 1, pp. 91–102, Jan. 1992.

- [44] I. Jackson, "Anonymous addresses and confidentiality of location," in *Proc. Int. Workshop Inf. Hiding*, Berlin, Germany: Springer, 1996, pp. 115–120.
- [45] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in GPS traces via uncertainty-aware path cloaking," in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*, 2007, pp. 161–171.
- [46] Z. Riaz, F. Durr, and K. Rothermel, "Location privacy and utility in geo-social networks: Survey and research challenges," in *Proc. 16th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2018, pp. 1–10.
- [47] R. Chow and P. Golle, "Faking contextual data for fun, profit, and privacy," in *Proc. 8th ACM Workshop Privacy Electron. Soc. (WPES)*, 2009, pp. 105–108.
- [48] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: Anonymizers are not necessary," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2008, pp. 121–132.
- [49] K. Sampigethaya and R. Poovendran, "A survey on mix networks and their secure applications," *Proc. IEEE*, vol. 94, no. 12, pp. 2142–2181, Dec. 2006.
- [50] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S. K. Nayar, "Face swapping: Automatically replacing faces in photographs," in *Proc. ACM SIGGRAPH Papers*, 2008, pp. 1–8.
- [51] P. Korshunov and T. Ebrahimi, "Using warping for privacy protection in video surveillance," in *Proc. 18th Int. Conf. Digit. Signal Process. (DSP)*, Jul. 2013, pp. 1–6.
- [52] X. Zhang, S.-H. Seo, and C. Wang, "A lightweight encryption method for privacy protection in surveillance videos," *IEEE Access*, vol. 6, pp. 18074–18087, 2018.
- [53] Y. Zarouk, I. Souici, S. Hallaci, and H. Seridi, "Evolutionary encryption algorithm ensuring privacy in video surveillance," *Int. J. Informat. Appl. Math.*, vol. 4, no. 1, pp. 28–55, 2013.
- [54] A. Fitwi, Y. Chen, and S. Zhu, "A lightweight blockchain-based privacy protection for smart surveillance at the edge," in *Proc. IEEE Int. Conf. Blockchain*, Jul. 2019, pp. 552–555.
- [55] G. R. Blakley, "Safeguarding cryptographic keys," in *Proc. Int. Workshop Manag. Requirements Knowl. (MARK)*, Jun. 1979, pp. 313–318.
- [56] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
- [57] L. Bai, S. Biswas, A. Ortiz, and D. Dalessandro, "An image secret sharing method," in *Proc. 9th Int. Conf. Inf. Fusion*, Jul. 2006, pp. 1–6.
- [58] L. Bai, "A strong ramp secret sharing scheme using matrix projection," in *Proc. Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, 2006, p. 5.
- [59] Z. Tu, K. Zhao, F. Xu, Y. Li, L. Su, and D. Jin, "Protecting trajectory from semantic attack considering k -anonymity, l -diversity, and t -closeness," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 1, pp. 264–278, Mar. 2019.
- [60] V. Vdurvceková, L. Schwartz, V. Hottmar, and B. Adamec, "Detection of attacks causing network service denial," *Adv. Mil. Technol.*, vol. 13, no. 1, pp. 1–9, 2018.
- [61] C. Esposito, Z. Zhao, and J. Rak, "Reinforced secure gossiping against DoS attacks in post-disaster scenarios," *IEEE Access*, vol. 8, pp. 178651–178669, 2020.
- [62] C. Benzaid, M. Boukhalifa, and T. Taleb, "Robust self-protection against application-layer (D)DoS attacks in SDN environment," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, May 2020, pp. 1–6.
- [63] G. Jaideep and B. Battula, "A framework for agent-based detection and prevention of DDoS attacks in distributed p2P networks," in *Pervasive Computing: A Networking Perspective and Future Directions*. Singapore: Springer, 2019, pp. 15–30.
- [64] G. DiAna, "Time-analysis of pseudorandom bit generators," *Math. Inst., Annales Universitatis Scientiarum Budapestinensis de Rolando Eötvös Nominatae, Budapest, Hungary, Tech. Rep.*, 2005, vol. 48.
- [65] B. Williams, R. E. Hiromoto, and A. Carlson, "A design for a cryptographically secure pseudo random number generator," in *Proc. 10th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst., Technol. Appl. (IDAACS)*, vol. 2, Sep. 2019, pp. 864–869.
- [66] S. Tochner, A. Zohar, and S. Schmid, "Route hijacking and DoS in off-chain networks," in *Proc. 2nd ACM Conf. Adv. Financial Technol.*, Oct. 2020, pp. 228–240.
- [67] D. J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Y. Halpern, "Worst-case background knowledge for privacy-preserving data publishing," in *Proc. IEEE 23rd Int. Conf. Data Eng.*, Apr. 2007, pp. 126–135.
- [68] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Enhancing security and privacy in traffic-monitoring systems," *IEEE Pervas. Comput.*, vol. 5, no. 4, pp. 38–46, Oct. 2006.
- [69] A. Konstantinidis, G. Chatzimilioudis, D. Zeinalipour-Yazti, P. Mpeis, N. Pelekis, and Y. Theodoridis, "Privacy-preserving indoor localization on smartphones," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 3042–3055, Nov. 2015.
- [70] J.-H. Song, V. W. S. Wong, and V. C. M. Leung, "Wireless location privacy protection in vehicular ad-hoc networks," *Mobile Netw. Appl.*, vol. 15, no. 1, pp. 160–171, Feb. 2010.
- [71] S. Shao, A. Khreishah, and M. Ayyash, "Evaluating the feasibility of random waypoint model for indoor wireless networks," *Internet Technol. Lett.*, vol. 4, no. 2, p. e214, 2020.
- [72] M. A. P. Chamikara, P. Bertok, I. Khalil, D. Liu, and S. Camtepe, "Privacy preserving face recognition utilizing differential privacy," *Comput. Secur.*, vol. 97, Oct. 2020, Art. no. 101951.



SERENA NICOLAZZO received the Ph.D. degree in information engineering from the University Mediterranea of Reggio Calabria, in 2016. From 2016 to 2017, she was a Research Fellow with the University Mediterranea of Reggio Calabria. Since 2019, she has been a Research Engineering Senior Engineer with the Architecture and KPI Group, Sky Italia. She is currently a Visiting Researcher with Middlesex University London. She also collaborates with Innovation Group and she is responsible for the design of architectures and key performance indicators for monitoring and troubleshooting of products, services, and tech platform. Her research interests include data science, security, privacy, and multi-social network analysis. She is also involved in several TPCs and editorial board of prestigious international conferences and journals in the context of data science and cybersecurity.



ANTONINO NOCERA received the Ph.D. degree in information engineering from the University Mediterranea of Reggio Calabria, in March 2013. He has been an Assistant Professor with The University of Pavia, since February 2019. His research interests include several research contexts, including security, privacy, social network analysis, machine learning, recommender systems, the Internet of Things, cloud computing, and blockchain. In these research fields, he published about 70 scientific articles. He is currently involved in several TPCs of prestigious international conferences in the context of data science and cybersecurity. He is also an Associate Editor of *Information Sciences* (Elsevier).



DOMENICO URSINO received the M.Sc. degree in computer engineering and the Ph.D. degree in system engineering and computer science from the University of Calabria, in July 1995 and January 2000, respectively. From January 2005 to December 2017, he was an Associate Professor with the University Mediterranea of Reggio Calabria. Since January 2018, he has been a Full Professor with the Polytechnic University of Marche. His research interests include source and data integration, data lakes, social network analysis, social internetworking, ecosystems consisting of the Internet of Things, innovation management, knowledge extraction and representation, biomedical applications, and recommender systems. In these research fields, he published more than 180 articles.

• • •