PriSE: Slenderized Privacy-Preserving Surveillance as an Edge Service

Conference Paper · December 2020

DOI: 10.1109/CIC50333.2020.00024

3 authors, including:

Alem Fitwi
Binghamton University
17 PUBLICATIONS **83** CITATIONS
SEE PROFILE

Yu Chen
Binghamton University
196 PUBLICATIONS **3,421** CITATIONS
SEE PROFILE

PriSE: Slenderized Privacy-Preserving Surveillance as an Edge Service

Alem Fitwi[a], Yu Chen[a], Sencun Zhu[b]
[a]Dept. of Electrical and Computer Engineering, Binghamton University, Binghamton, NY 13902, USA
[b]Dept. of Computer Science and Engineering, Penn State University, University Park, PA 16802, USA
Emails: {afitwi1, ychen}@binghamton.edu, sxz16@psu.edu

2020 IEEE 6th International Conference on Collaboration and Internet Computing (CIC) | 978-1-7281-4146-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/CIC50333.2020.00024

***Abstract*—With a myriad of edge cameras deployed in urban areas, many people are seriously concerned about the invasion of their privacy. The edge computing paradigm allows enforcing privacy-preserving measures at the point where the video frames are created. However, the resource constraints at the network edge make existing compute-intensive privacy-preserving solutions unaffordable. In this paper, we propose slenderized and efficient methods for Privacy-preserving Surveillance as an Edge service (PriSE) after investigating a spectrum of image-processing, image scrambling, and deep learning (DL) based mechanisms. At the edge cameras, the PriSE introduces an efficient and lightweight Reversible Chaotic Masking (ReCAM) scheme preceded by a simple foreground object detector. The scrambling scheme prevents an interception attack by ensuring end-to-end privacy. The simplified motion detector helps save bandwidth, processing time, and storage by discarding those frames that contain no foreground objects. On a fog/cloud server, the scrambling scheme is coupled with a robust window-detector to prevent peeping via windows and a multi-tasked convolutional neural network (MTCNN) based face-detector for the purpose of de-identification. The extensive experimental studies and comparative analysis show that the PriSE is able to efficiently detect foreground objects and scramble frames at the edge cameras, and detect and denature window and face objects at a fog/cloud server to ensure end-to-end communication privacy and anonymity, respectively. This is done just before the frames are sent to the viewing stations.**

***Keywords*-Edge Computing, Privacy Preserving, Reversible Scrambling, Window-Face Detector.**

I. Introduction

Real-time information fusion and situational awareness (SAW) are essential for smart cities [38], [39]. With the widespread use of surveillance cameras, privacy is becoming an increasingly important issue. Hence, surveillance systems are expected to have the capability to prevent the spill of collected information to the cyberspace owning to interception attacks. Furthermore, most of the preexisting mechanical surveillance systems rely on security personnel to observe any illicit acts, and to perform analysis of the captured videos whenever deemed necessary [20]. As a result, authorized security personnel in charge of the surveillance system might abuse the cameras for voyeurism, peeking via windows, cyber stalking, or unauthorized collection of data on activities or behaviors of individuals [8], [31], [37], [40]. As the surveillance cameras are equipped with better optical and digital zooms, the more likely they are to be abused to gather private information unless preventive measures are enforced by design.

Edge-computing paradigm creates auspicious environment for enforcing privacy-preserving measures at the point where the video is created. The edge environment supports slenderized techniques that require less computational resources. Hence, exploiting the capabilities of the edge computing paradigm, a lightweight object scanner and scrambler modules can be developed to ensure end-to-end privacy and to anonymize individuals filmed on cameras.

In this paper, we propose a slenderized Privacy-preserving Surveillance as an Edge service (PriSE). Following the edge-fog/cloud computing paradigm, the PriSE scheme introduces a lightweight frame scrambling method that ensures end-to-end privacy preservation. While human faces are the most powerful features to identify people, windows have been used for long as inlets for spying individuals [30] at their homes. The high-profile incidence publicly known is the spying on the private Berlin flat of the German Chancellor Angela Merkela [4] by a CCTV camera directed through her window from a Museum. Focusing on windows and human faces, therefore, the PriSE scheme comprises a customized window-object detector and a multi-tasked convolutional neural network (MTCNN) based face detector. Windows and faces are detected and denatured to prevent peeping and to ensure de-identification, respectively. The PriSE system enables detection of wanted human objects such as known criminals and their faces are left untouched. In practice, it is a challenging task for both accuracy and performance reasons when lists of wanted people increase.

The major contributions of this paper are summarized as follows:

- A novel optimized and lightweight Reversible Chaotic Masking (ReCAM) technique is proposed for scrambling frames/images to ensure an end-to-end privacy. The ReCAM scheme is more suitable for image scrambling at the edge than traditional data encryption schemes and existing chaotic schemes. It is a faster scheme whose Its security and performance are proved based on standard security and performance metrics, respectively.
- A simplified motion detection algorithm is introduced at the edge that discards frames with no new foreground objects. A customized window-detector detects and masks window-objects on frames to prevent prowling through them by maneuvering the cameras. An MTCNN based face detector detects faces on frames for a subsequent face de-identification process.
- Extensive experimental studies were carried out,

978-1-7281-4146-6/20/$31.00 ©2020 IEEE
DOI 10.1109/CIC50333.2020.00024

including performance analysis and a comparison with existing equivalent methods. The results corroborate that the PriSE scheme provides an end-to-end privacy protection against possible interception attacks and it is able to detect and denature window and face objects to prevent unauthorized data collection and identification of individuals, respectively.

The remainder of this paper is organized as ensues. The related works are presented in Section II. Section III describes the architectural overview of the proposed PriSE scheme. A computationally-efficient and secure frame-scrambling technique is explained in Section IV. Section V explains how window and face objects are detected and masked to prevent peeping and unauthorized identification followed by the brief presentation of the key distribution management technique in Section VI. The details of experimental setup, discussion of comparative security and performance analysis, and corresponding results are elucidated in Section VII. At last, our concluding remarks are presented in Section VIII.

II. Related Works

A. Privacy in Surveillance Systems

CCTV cameras are nowadays ubiquitously deployed in urban areas by governments, companies, and individuals to ensure physical security and safety. For example, an average Londoner is estimated to be caught on CCTV cameras 300 times a day while moving around the city [8], [9], [37], [41]. As a result, a huge amount of information about individuals is collected without their knowledge and consent. Hence, the best solution to protect the privacy of individuals amid the increased proliferation of fixedly deployed cameras and drones is demanding the camera and drone companies to incorporate privacy-preserving methods by design [2], [5], [7], [35].

The invasion of privacy in surveillance systems is often attributed to two major factors: interception of videos while on transit and abuse of CCTV's by the operators. Most existing surveillance systems are deployed based on the cloud computing architecture. As a result, videos could be intercepted while being transported over the network exploiting some of its vulnerabilities [3], [9], [7], [18], which may cause the spilling of private data into the cyber space. Secondly, people in charge of the CCTV cameras could illicitly collect private data. Additional abuses of CCTV cameras, as identified by the American Civil Liberties Union (ACLU), include criminal abuse, institutional abuse, abuse for personal purposes, discriminatory targeting, and voyeurism [27]. However, apart from pinpointing the privacy weakness of the surveillance system, none of these papers propose a viable solution.

B. Personal Privacy Attributes

The personal privacy attributes refer to the privacy-sensitive classes of objects for which individuals demand protection. They are the contents on the video frames and images captured and conveyed by the surveillance systems that can reveal a lot of personal information about the identity, behavioral pattern, and daily engagements of individuals [27], [34], [40]. Attributes like face reveal substantial details for identifying individuals. Besides, transmitting nude frames as-is could cause social embarrassment if intercepted. People also feel nervous to think that they would be filmed by cameras while visiting some places like clinic. In addition, people do not want to be furtively observed via openings like windows while at home. These works attempt to define and identify privacy-sensitive objects but they do not have privacy-preserving mechanisms. A very good machine-learning-based privacy setting recommendation system for image sharing on social media was proposed in [40]. However, it gives no information how the privacy of the images of social media users is protected while being sent to a cloud server for recommendation. Besides, there are no robustly designed schemes to tackle the problem of peeping via windows carried out by maneuvering CCTV cameras apart from stating it as a longstanding privacy problem.

C. Advancements in Object-Detection Methods

Object classification and localization is vital in the effort to create privacy-preserving surveillance systems. There are a number of convoluted networks like VGG-Net [29], Res-Net [12], and many others [6], [17], [32] that primarily focus on improving accuracy using deeply involved networks. But the recent trend of research in the area of CNN-based object-detection is directed towards designing, developing and deploying compact networks [13], [14], [28] for mobile applications. However, their performance on edge devices like Raspberry PI is not impressive yet. All of the publicly available lightweight deep neural networks (DNN) like GoogLeNet, SqueezeNet, MobileNet version 1, and MobileNet version 2 can only process less than 4 colored 480P frames per second (fps) on a Raspberry PI 4.

With regards to face recognition and detection, there are many popular methods with nearly human accuracy developed over the last two decades. The first popular method is a Haar Cascade Face Detector [36]. It is a relatively faster method with simple architecture that can detect faces of different scales. It, however, suffers from a little bit higher false positive rates (FPR) and terribly poor performance under occlusion. Besides, it detects only frontal faces. Another widely used face detection model is based on the histogram oriented gradient (HOG) features and support vector machine (SVM) [21]. It is the fastest known method on CPUs. Besides, it works very well for frontal and slightly non-frontal faces under small occlusion. But it fails to detect smaller faces, side faces, and does not work very well under substantial occlusion. In 2015, Google developed a DNN based robust FaceNet [25] that achieves 99.96% accuracy rate on the facial recognition dataset Labeled Faces in the Wild (LFW). The most popular and most accurate face detection method today that attempts to addresses both face alignment and detection problems arising from occlusion and illumination is a Multi-Tasked Cascaded Convolutional Neural Network (MTCNN) [15], [42]. It is a DNN based architecture that comprises three neural networks (Proposal Network (P-Net), Refine Network (R-Net), and Output Network (O-Net)) connected in a cascaded fashion.

Furthermore, there were specific attempts to leverage the advancement in machine learning and object-detection technologies to localize privacy-sensitive objects [1], [19], [31], [37], [40]. They give some idea about how to build privacy-aware real-time video analytics in the surveillance systems.

D. Video Frame Enciphering Schemes

There are a number of image scrambling techniques [10], [16], [33] that could be employed to scramble frames. But they need to be made lighter as to fit into a resource-constrained environment. Besides, they must achieve a good balance between privacy, clarity, reversibility, security, and robustness [22]. Encryption is the most secure member of the editing class privacy protection schemes. It protects private information and sensitive data from unauthorized accesses, and enhances the security of communication between two communicating parties. However, given the real-time nature of videos, the traditional symmetric key cryptographic mechanisms like Advanced Data Encryption (AES) are not convenient for image scrambling, mainly due to low processing speed. Chaotic-encryption mechanism offers better solutions in that they have better performance, high degree of sensitivity to slight changes in initial conditions, higher degree of randomness, enormous key space, aperiodicity, and high security. Recently proposed chaotic image enciphering mechanisms [16], [33] are secure but slower to be employed at the edge.

III. System Architecture of the PriSE

In this paper, we propose a slenderized Privacy-preserving Surveillance as an Edge service (PriSE) following the edge-fog/cloud computing paradigm. The PriSE system performs three major tasks, namely foreground object detection, frame scrambling, and face and window objects detection and masking. The video frame scrambling process is carried out at the network edge preceded by a simple foreground object scanning. That is, on the cameras, a thin and simplified motion detector detects motion of an object or objects in order to discard frames that contain only background frames. All object-containing frames are scrambled and sent to the video analytics center at a fog/cloud server, where an inverse scrambling process is performed followed by window and face objects detection and masking processes. Detecting windows and scrambling them helps prevent peeping via windows, and detecting and masking faces of individuals caught on camera ensures de-identification. Figure 1 illustrates the major tasks of PriSE and the order in which they are executed starting at the edge camera up to the surveillance operation center (SOC). They are summarized as ensues and detailed explanations are furnished in subsequent sections.

1. Simplified Motion Detector: Video streams captured by edge cameras (wall mounted or those perched on poles) have mostly relatively static or unchanging background over a series of video frames. Hence, it is possible to model and monitor the background for significant changes that indicate object motions captured by the camera. There are many sophisticated foreground and background segmentation and subtraction techniques [11], [43] for motion detection. However, they are computationally expensive and infeasible for edge implementation.

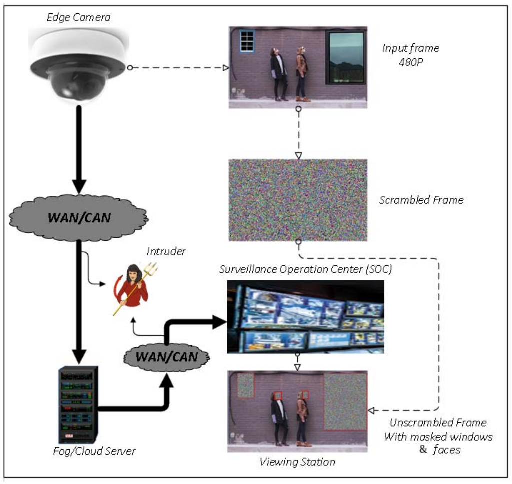

Fig. 1. Architecture of the PriSE system comprising Edge camera that creates and encrypts frames, intruder that tries to access the video stream on transit, cloud/fog server that deciphers frames, detect windows and faces and mask them, and a viewing station.

Algorithm 1 shows a lightweight motion detector proposed for edge implementation. The lightweight motion detector reduces latency and saves bandwidth and storage by discarding a number of frames with no objects or with redundant information. The first frame is used as a reference frame ($frame_{ref}$) updated every hour to account the impact of environmental factors that affect the light intensity. Then, the percentage of the difference ($frame_{diff}$) between every frame created and the reference frame is computed using a fast vector operator. The output is discriminated into black (background) and white (foreground) using a binary threshold. Then, a minimum percentage of nonzero pixels (nz_pixels) in the difference frame is selected, which determines whether or not to trigger the scrambling and transmission of the current frame. In our study, the threshold is experimentally chosen to be 0.29%. Frames that have less than 0.29% ($nz\_pixels < 0.29\%$) nonzero pixels are discarded. This value is made smaller to make sure that no real object is skipped undetected.

2. Frame Privacy-preserving Scheme: this is the backbone of the proposed PriSE scheme. Primarily the privacy of individuals could be affected either when the collected raw information is intercepted while on transit or when cameras are abused by people in charge, like peeping via windows. This paper addresses the first problem by introducing a novel 2D lightweight Reversible Chaotic Masking (ReCAM) technique. Every frame that contains an object or objects other than the background object is encrypted at the edge and sent to a remote server connected to a surveillance operation or viewing center. On the server, the unscrambling process is carried out to uncover the object-containing plain frames for subsequent processing.

3. Window and Face Objects Detection and Masking: This method is introduced to stop the collection of private information and indoor activities of individuals through the

Algorithm 1 @ Edge Camera

```
t_0 ← system_time()
flag ← 0
vid ← videoCapture()
frame_Ref ← None
W, H ← 640, 480 (frame dimensions)
while True do
    status, frame ← vid.read()
    frame ← frame.resize((W, H))
    if status then
        frame_Gray ← to_gray(frame)
        frame_Gray ← GaussianBlur(frame_Gray)
        t_1 ← system_time()
        t_2 ← $t_1 - t_0$
        if (frame_Ref == None) or (t_2 == 1 hour) then
            frame_Ref ← frame_Gray
            flag ← 1
            if t_2 == 1 hour then
                t_0 ← $t_1$
        frame_Diff ← vectorized_xor(frame_gray, frame_Ref)
        frame_Diff ← binary_threshold(frame_Diff)
        pixels ← array(frame_Diff)
        nz_pixels ← $\frac{count\_nonzero(pixels)}{H \times W}$
        if nz_pixels $\geq$ 0.0029 or flag == 1 then
            if flag == 1 then
                frame ← scramble(frame_Ref)
                flag ← 0
            else
                frame ← scramble(frame)
        else
            continue
```

notorious practices of furtively peeping through windows and face identification. In PriSE, frames are checked if they contain windows and faces using pertinent object-detectors. If they do, the windows and faces are reversibly masked at the server before they are forwarded to the viewing station to prevent peeping and identification by face, respectively.

IV. Reversible Privacy-Conserving Scheme

To at least minimize the risk of privacy invasions, individuals caught on CCTV cameras need to be anonymized. The human face is the most powerful identifying human feature and it needs to be scrambled for the purpose of de-identification. Besides, the people in charge of the cameras should not be allowed to unauthorizedly prowl through windows by maneuvering the cameras to observe people at home. This section focuses on designing an efficient and lightweight scrambling technique for preserving the privacy of both full frames and regions of interest (face and window) on a frame. A 2D lightweight Reversible Chaotic Masking (ReCAM) is introduced based on a dynamic chaotic system, built from a homogeneous second order differential equation of the form given in Eq. (1), whose solution has unique complex roots.

$$a\frac{d^2x}{dt^2} + b\frac{dx}{dt} + cx = f(x) \tag{1}$$

The obtained 2D chaotic solution is shown as Eq. (2). Due to the limited space, the deduction steps are skipped.

$$\begin{aligned} x(t) &= ((7.4 \times 2.7^{\lambda t}[Acos(\mu t) + Bsin(\mu t)])\%1) \\ y(t) &= ((7.4\lambda \times 2.7^{\lambda t}[Ccos(\mu t) + Dsin(\mu t)])\%1) \end{aligned} \tag{2}$$

To efficiently generate a secure chaos, the values of the control parameters in the system of equations for $x(t)\ and\ y(t)$ in Eq. (2) must always be within their respective domains determined experimentally and provided in Table I.

TABLE I
Control Parameter Specifications

Parameters	Secure Domain
λ	$\lambda \in [0.0005, 0.0009]$
μ	$\mu \in [0.999999594999918,$ $0.9999998749999922]$
A	$A \in [1, 256]$
C	$C \in [256, 512]$
B	$B = k_1 \times \frac{C-\lambda A}{\mu}$, $k_1 \in [0.89, 0.999999]$
D	$D = k_2 \times \frac{\lambda C-\lambda^2 A-\mu^2 A}{\mu}$, $k_2 \in [0.9, 0.999999]$

The control parameters provided in Table I not only affect the output chaos content but also its security; the chaos generator is not equally sensitive to all of them, though. Any change in the values of any of the eight parameters (λ, μ, A, B, C, D, k_1, & k_2) produces a completely different 2D chaos. However, the λ value is the one that can have tremendous effect on the security of the chaos. The other parameters have relatively smaller impact on the security. But all of them are used as component of the scrambling key.

Figure 2 demonstrates that the first three scatter plots, colored green, are secure because the values of λ used to generate the chaotic outputs are within the range of the secure domain defined in Table I. However, the one in red color is not a good chaos; it has some patterns. This is due to the fact that the value of λ employed is 0.01, which is out of the secure domain. Parameters A & B can assume any positive numeric value as big as the computer where this module runs can support, as long as the value of λ is within the secure range. However, for computational efficiency, the values of parameters A and B are constrained to domains $[1,\ 255]$ and $[256,\ 512$, respectively. The variable t can assume a value as big as the size of the 2D-chaos intended to be generated, $t \in [0, W \times H - 1]$, where H &W are, respectively,the height and width of the chaos.

Pixel values of red (R), green (G), and blue (B) channels of the chaos used for scrambling a frame are generated using different set of keys as Eq. (3) shows. A secure 2D-chaos is produced for each color channel using the corresponding keys.

$$\begin{aligned} Key\ &=\ [K_R,\ K_G,\ K_B] \\ K_R &= [\lambda_r,\ \mu_r,\ A_r,\ C_r,\ B_r,\ D_r,\ k_{1r},\ k_{2r}] \\ K_G &= [\lambda_g,\ \mu_g,\ A_g,\ C_g,\ B_g,\ D_g,\ k_{1g},\ k_{2g}] \\ K_B &= [\lambda_b,\ \mu_b,\ A_b,\ C_b,\ B_b,\ D_b,\ k_{1b},\ k_{2b}] \end{aligned} \tag{3}$$

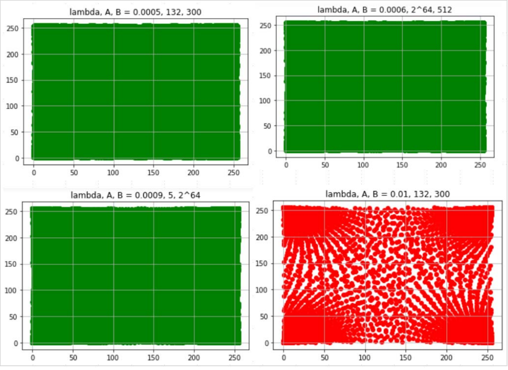

Fig. 2. The effects of parameters' values on the security of the output chaos.

This scrambling module is then incorporated to the simple motion-detection method portrayed in Algorithm 1 at the edge and the scrambling process is performed as briefly described in Algorithm 2.

Algorithm 2 Frame Scrambling and its Inverse Process

```
@ Edge Camera/Sensing End
$frame_{input} \leftarrow vid.cam()$
$H,\ W \leftarrow frame_{input}.size()$
procedure GENERATEKEY
    $key \leftarrow secRand([\lambda,\ \mu,\ A,\ C,\ B,\ D,\ k_1,\ k_2])$
    return key
procedure PRODUCECHAOS(key, H, W)
    $chaos \leftarrow$ (key, H, W)
    return chaos
procedure SCRAMBLE(chaos, $f_{input}$)
    $frame_{encrypted} \leftarrow frame_{input}\ xor$ chaos
    return $f_{encrypted}$
@ Receiving End
$key \leftarrow$ received from sending end
$frame_{encrypted} \leftarrow$ received from sending end
$chaos \leftarrow produceChaos(key,\ H,\ W)$
procedure UNSCRAMBLE(chaos, $frame_{encrypted}$)
    $frame_{decrypted} \leftarrow frame_{encrypted}\ v.xor$ chaos
    return $frame_{clear}$
```

V. Window and Face Detection and Denaturing

Object classification and localization helps identify sensitive parts of interest on video frames. This section focuses on DNN based models for window and face object detection, where the main process flows are portrayed in Fig. 3.

Window objects consist of a number of edges that show significant changes in pixel intensity, lines (1-D structures) between image regions with same intensity on either sides, partitioned rectangular or approximately rectangular shapes, and arcs or semicircles. They could be easily detected using a combination of filtering, thresholding, edge-detection, segmentation, and contouring methods. Such approach is lightweight and faster but it suffers from a relatively higher false positive rate.

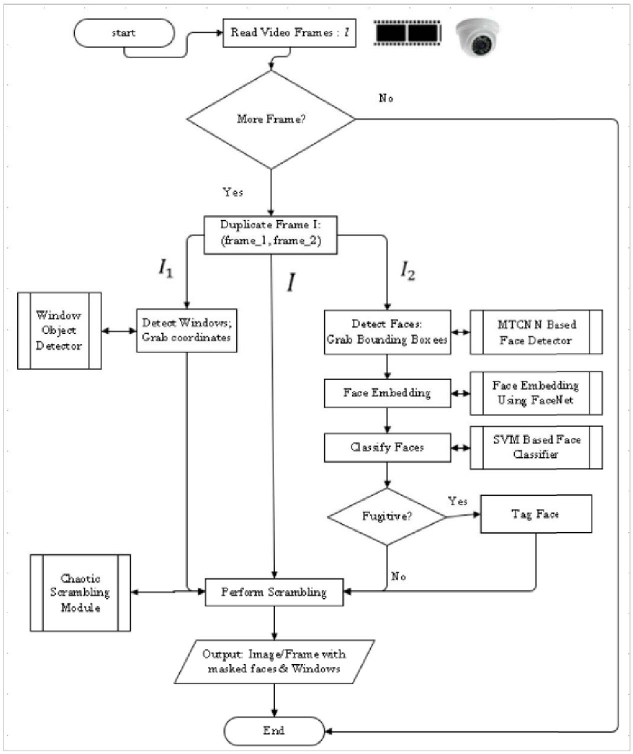
start
Read Video Frames : I
More Frame?
No
Yes
Duplicate Frame I: (frame_1, frame_2)
I_1
I
I_2
Window Object Detector
Detect Windows; Grab coordinates
Detect Faces: Grab Bounding Boxees
MTCNN Based Face Detector
Face Embedding
Face Embedding Using FaceNet
Classify Faces
SVM Based Face Classifier
Fugitive?
Yes
Tag Face
No
Chaotic Scrambling Module
Perform Scrambling
Output: Image/Frame with masked faces & Windows
End

Fig. 3. Window and Face object detection processes.

To train and create window-detector model, Tensorflow based on MobileNetv2 [24], we created a dataset comprising 27,000 window images downloaded and web-scrapped from the Internet because there is not a publicly available dataset for windows. Sixteen types of windows of different types, orientations, style, color intensity, and material are considered.

Robust object detection method is essential for building a good privacy preserving mechanism. Hence, a robust face detector model based on the state-of-the-art MTCNN model is trained [15], [42]. It can detect and extract faces in different orientations (left, up, down, and right), even under some occlusions. As shown in Fig. 3, the face processing includes the following models:

1. MTCNN based Face Detector: The MTCNN employs a 3-stage cascaded framework. The first stage is a P-Net that estimates the bounding box regression vectors to calibrate the candidate faces after which non-maximum suppression (NMS) is applied to put highly overlapped faces together. The R-Net is the second stage that refines false faces and carries out the bounding box regression calibration and NMS candidate merging. The O-Net is the third stage that describes the face in a more detailed manner. It outputs the five facial landmarks' positions. This way, the MTCNN addresses the alignment problems in many other face detection algorithms. We built an MTCNN-based face-detector using TensorFlow and trained it using images from the Challenge of Recognizing One Million Celebrities in the Real World and an open source [26].

2. FaceNet [25]: A deep CNN based unified embedding for Face detection and Clustering that uses a triplet loss function for training. The triplets loss function is calculated from the triplet of three pictures, Anchor (A), Negative (N), and Positive (P) images. Its main goal is to distinguish positive and negative classes by distance boundaries. It extracts high-quality 128 element vector features from the face used for future prediction and detection of faces. We employed it to create face embedding.

3. SVM Model: A Linear SVM model is developed to classify face embeddings as one of the wanted criminals or fugitives. The faces of wanted criminals or fugitives from justice are first detected and extracted using MTCNN followed by face embedding performed by the FaceNet model. Then, the face embedding vectors are normalized and fit into the SVM model. Hence, every incoming frame is tested using the trained SVM model to check whether it contains faces of wanted people. If a face is classified as wanted, it is tagged and is not denatured.

After creating window and face objects detectors, all inferences are performed on the cloud/fog server. While the window-and-face detection and denaturing processes can be performed at the edge, doing so consumes resources. In our design, every object-containing frame is encrypted at the edge and transmitted to the receiving end. Hence, detecting the windows or faces, denaturing them and then scrambling them along with other frame contents again during frame enciphering at the edge is unnecessarily costly. Hence, the approach pursued is that frames are transmitted from the edge cameras to the fog/cloud server in fully encrypted form. Then, the detection of window-and-face objects and subsequent denaturing are performed on the server just before they are forwarded to the viewing stations as portrayed in Fig. 1.

VI. Key Distribution

As portrayed in the system architecture in Fig.1, video streams are transmitted from cameras to the server, and then from the server to the viewing stations in encrypted form. Hence, a client-server architecture based lightweight agents are employed for efficiently managing the key distribution as shown in Fig. 4. The key is generated in the form of list data structure, $Key = [K_R, K_G, K_B]$, where K_R, K_G and K_B are the keys for the three channels of an input frame. The camera agent stores the public key of the server agent, and generates a session key, as described in Algorithm 2, for every frame scrambling. The server agent keeps record of the public keys of all clients for the purpose of secure session key exchange. The viewing-station agent unscrambles frames to be viewed live by security personnel in a SOC. It keeps the public key of the server for recovering a session key that is used to decipher encrypted frames forwarded by the server.

VII. Experimental Study

In our PriSE system, smart cameras at the edge are equipped with Raspberry Pi 4 boards, whose detailed specifications are provided in Table II. The implementation is done using Python 3.7.4 multithreading and multiprocessing, where the global interpreter lock (GIL) is disabled. In addition, the video frame/image used for the experiment and implementation has a size of 480P (480x640x3) and three color channels (RGB) with 8-bit wide pixels.

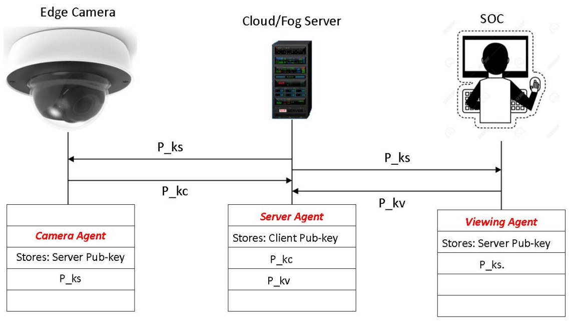

Fig. 4. Client-server based key distribution management.

TABLE II
Specification of the Raspberry Pi 4

Parameters	Specifications
CPU type/speed	*Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz*
RAM size	*4GB LPDDR4-2400 SDRAM*
Integrated Wi-Fi	*2.4GHz and 5GHz*
Ethernet speed	*1Gbps*
Camera port	*2-lane MIPI CSI*
Bluetooth	*5.0*
Power Requirement	*3A, 5V*
Operating System	*Debian Linux 10 based*

A. Simple Motion Detector Algorithm

Figure 5 demonstrates how the simple motion detection algorithm works using video frames captured by a camera with Raspberry PI. Figure 5(a) is employed as a reference against which subsequent frames are compared to check if there is any motion. The next frame in Fig. 5(b) is similar to the reference frame. As a result, their difference becomes null or a black frame as illustrated in Fig. 5(c). However, the frame in Fig. 5(d) contains a fast-moving object. Consequently, the difference between the reference frame and the frame in Fig. 5(d) is significant, as illustrated in Fig. 5(e). It effectively detected the object even though it moved at a speed more than 200 feet per second.

All frames whose difference with the reference frame has at least 0.29% nonzero pixels are scrambled and transmitted; otherwise, discarded. This value is made smaller to make sure that no real object is skipped undetected. Less than 0.29% pixel changes in the difference frame are often attributed to environmental factors. To analyze its benefit in terms of bandwidth, storage, and processing time, we considered a video with 750 frames where objects are captured only in 135 of them. In the rest of the 615 frames, only the background object is captured. Hence, the 615 frames are discarded and only the 135 are encrypted and forwarded to the server. This saves about 120s of processing delay and 184MB of memory. Besides, our scheme coupled with the scrambling module can process more than 5 frames per second (fps). However, if standard foreground object detectors [11], [43] are employed at the edge, only about 1 fps can be processed.

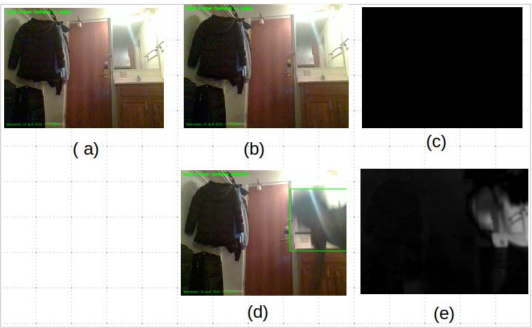

Fig. 5. Raspberry PI-camera-2 Motion Detection:(a) reference frame, (b) next frame with no change, (c) a null frame which is the difference of (a) and (b), (d) frame containing a fast-moving person (e) a frame containing only the difference of (a) and (d)

B. Security Analysis of the Scrambling Method

In this section, the function, performance and security of the proposed video frame/image scrambling technique, ReCAM, are analyzed in detail. A scrambling scheme is said to be efficient if it has high exhaustive-key-search time, good security and low computational time complexity. Hence, a number of parameters, standard criteria, and test suites have been employed to evaluate it. US National Institute of Technology (NIST) randomness test suite, time complexity, key space analysis, histogram analysis, key sensitivity, Pixel sensitivity, Peak Signal to Noise Ratio (PSNR), Number of Pixels Change Rate (NPCR), Unified Average Changing Intensity (UACI), Information Entropy, horizontal correlation, vertical correlation, and diagonal correlation are considered.

Our ReCAM scheme is compared to three other image encryption methods selected based on speed, their being state-of-the-art, and security. One is AES, the most secure and most widely used symmetrical encryption standard in today's Internet TLS. The other two are chaotic-based encryption schemes proposed by Liu [16] and Tang [33], respectively. Liu's solution is a relatively lighter scheme designed based on a parameter-varied non-linear logistic chaotic map, the simplest and well-studied chaotic map. It offers good speed and security unlike other higher dimension chaotic systems. The Liu's shuffling round considered in our experiments is $T = 2$. Tang's solution is one of the most recently published chaotic encryption schemes. It is secure and designed based on low complexity of double spiral scans and a chaotic map.

1) Visual Assessment: The cipher image of a good scrambling technique must not give any visual information about the corresponding plain input image. Figure 6 illustrates an input image and a video frame input along with their corresponding ciphers. Neither ciphers give visual clue about their corresponding plain input images. Hence, the proposed scheme is secure against visual analysis attacks.

2) NIST Randomness Test Suite: The US NIST has established a test suite to evaluate the randomness of random number generators for cryptographic purposes [23]. It is a statistical package comprising 15 tests. The ReCAM has a high degree of randomness and passed all the NIST randomness tests as summarized in Table III. Strong randomness implies that the encrypted frame gives no clue about its clear version.

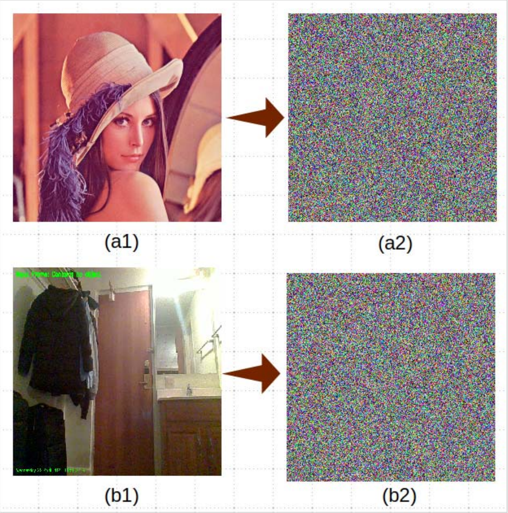

Fig. 6. Visual Assessment (a1) & (b1) are plain images and (a2) & (b2) are respectively their scrambled versions.

TABLE III
15 NIST Randomness Test Results

Tests	P_value	>= 0.01?
Monobit	*0.532*	*Yes*
Block Frequency	*0.172*	*Yes*
Run	*0.171*	*Yes*
Long Run	*0.842*	*Yes*
Matrix Rank	*0.162*	*Yes*
Spectral	*0.243*	*Yes*
Nonoverlapp'g Template Match	*0.052*	*Yes*
Overlapping Template Match	*0.241*	*Yes*
Maurer's Universal	*0.999*	*Yes*
Linear Complexity	*0.184*	*Yes*
Serial	*0.502*	*Yes*
Approximate Entropy	*0.602*	*Yes*
Cumulative Sum	*0.598*	*Yes*
Random Excursion	*0.667*	*Yes*
Random Excursion Variant	*0.562*	*Yes*

3) Statistics Comparison: Table IV compares our ReCAM scheme with three benchmark schemes. Normally an 8-bit image comprises pixel values that range from 0 to 255 whose frequencies vary from 0 to the order of thousands depending on the image size. In an ideal situation where the pixel values of an RGB image are assumed to be equally distributed, the image is supposed to have the statistics depicted in Table V, computed using the pythonic-numerical statistical methods. Table IV shows that the statistics of the various methods including ours are very close to the ideal values, which signify good distribution of the pixel values.

4) Histogram Analysis: Histogram analysis gives the frequency description of each unique pixel values (0 to 255) of an image. As a security requirement, the frequency of every unique pixel value is expected to be nearly equal.

TABLE IV
Comparative Security and Performance Analysis

Statistics	ReCAM	Liu's	Tang's	AES
Count	921600	921600	921600	921600
Mean	127.499	127.893	127.743	127.31
STD	*73.897*	*73.863*	*73.784*	*73.912*
Min	*0*	*0*	*0*	*0*
25%	*63*	*64*	*63*	*63*
50%	*127*	*128*	*127*	*127*
75%	*191*	*191*	*192*	*191*
Max	*255*	*255*	*255*	*255*

TABLE V
Ideal Statistics of an Image with uniformly distributed pixels

Mean	STD	Min	25%	55%	75%	Max
127.5	**73.901**	**0**	**63.75**	**127.5**	**191.25**	**255**

Figure 7 shows the histograms of the three color channels of the Lenna picture in clear form, which obviously are not uniform. Meanwhile, the frequencies of the color channels of the ReCAM cipher in Fig. 8 and the AES cipher in Fig. 9 are uniformly distributed. This verifies that the scrambling schemes are robust against any statistical histogram attacks in that the scrambled images have uniform histogram. And the histogram of our ReCAM scheme is the best.

5) Comparative Security Analysis: Our ReCAM scheme is also compared with the benchmark techniques using other parameters. Key space tells the robustness against exhaustive-search analysis. The lower boundary of a secure key space of a symmetrical encryption scheme is often considered to be 2^{128}. The key sensitivity measures how much the cipher changes when the key is slightly changed, in this case by only a bit. It is measured in terms of NPCR and UACI. A scheme with NPCR over 99% and UACI over 33% is considered to be secure against differential attacks. Unlike the measure of the quality of reconstruction of lossy compression, the PSNR of a good scrambling scheme is expected to be lower. The information entropy measures the amount of randomness in the information content of the scrambled image containing N pixels. The ideal value of entropy for an 8-bit pixel image I is $H(I) = 8$. Hence, a scheme with an entropy value close to eight is secure against entropy attack. The last criteria is the time complexity, measured in terms of the number of frames encrypted per second (FPS). So, the number of frames encrypted per second (FPS) is used to compare the schemes. The higher the FPS, the better scheme is in terms of speed.

Table VI shows that all the methods have comparable results in terms of key space, key sensitivity, PSNR, and entropy. In addition, our scheme has sufficiently large key space. If we assume that the exaFLOPS computer can perform 10^{15} decryptions per μs, then our scheme can be broken in 2.3×10^{106} years by brute force analysis using this machine. However, the ReCAM scheme stands out in terms of the cipher correlation in horizontal, vertical, and diagonal directions, and frame process speed. Our ReCAM scheme is designed to be lightweight and is much faster than the other schemes as shown in Table VI and Fig. 10. Figure 10 also shows that the FPS decreases as the frame size increases. For better processing speed and efficient bandwidth management, often smaller frame sizes like 480P and 720P are used in the practice of surveillance.

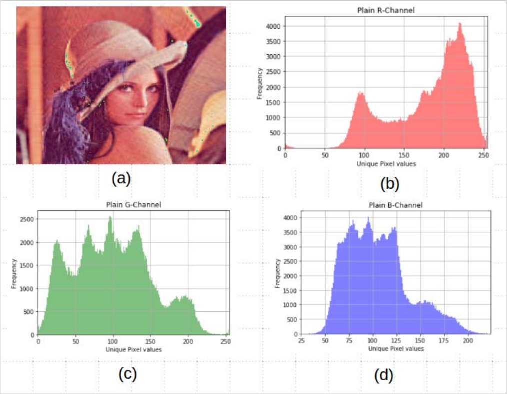

Fig. 7. Histogram of a plain image: (a) A plain RGB-image of Lenna, (b) histogram of R-channel of the image, (c) histogram of G-channel of the image, and (d) histogram of B-channel of the image.

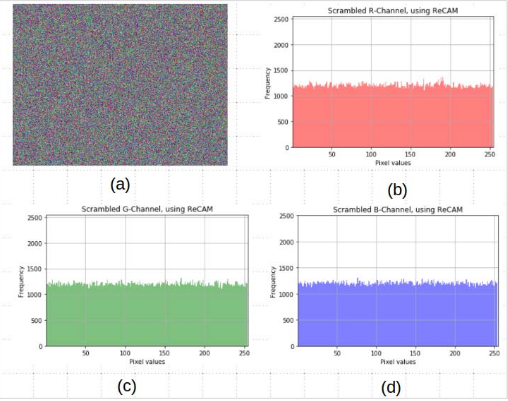

Fig. 8. Histogram of the RGB-channels of the cipher of our scheme (ReCAM).

The scheme in [9] focuses on minors detection and masking. It employs a minor's face masking scheme based on improved De Jong scheme. Its vectorized multiplication operator is specially designed for scrambling region of interest and it is slower if used for full frame scrambling.

C. Denaturing Window and Face Objects

To prevent prowling through windows to observe and record people in their homes by someone in charge of the cameras, all detected windows are denatured before frames are forwarded to the surveillance operation centers. The customized model is able to successfully detect more than 99.3% of the various of windows tested due to their simple natures. Besides, faces of individuals caught on cameras are denatured for the purpose of de-identification. Faces are detected with an average accuracy of 99.5%. But faces classified as wanted are left untouched.

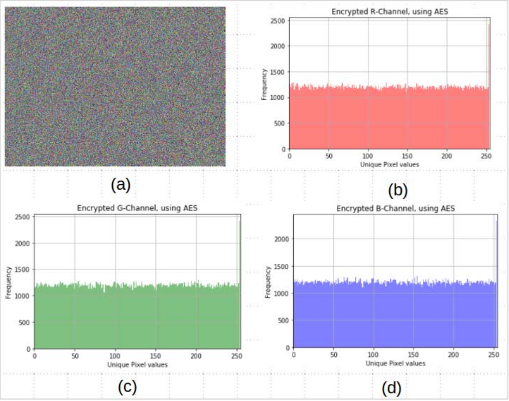

Fig. 9. Histogram of the cipher of AES.

TABLE VI
COMPARATIVE SECURITY AND PERFORMANCE ANALYSIS

Parameter	ReCAM	Liu's	Tang's	AES
Key Space	2^{448}	2^{2183}	$\approx 2^{407}$	2^{256}
Key Sensitivity				
UACI	33.456%	33.381%	33.379%	33.478%
NPCR	99.673%	99.657%	99.642%	99.63%
PSNR	9.43dB	11.231dB	10.73dB	9.102dB
Entropy	7.998 bits	7.999 bits	7.999 bits	7.999 bits
Horizontal Correlation	0.0006	0.0045	-0.00485	0.021
Vertical Correlation	0.0009	0.0039	0.0643	0.0067
Diagonal Correlation	0.0003	0.0054	0.0035	0.0029
FPS	5.61	1.215	0.346	0.722

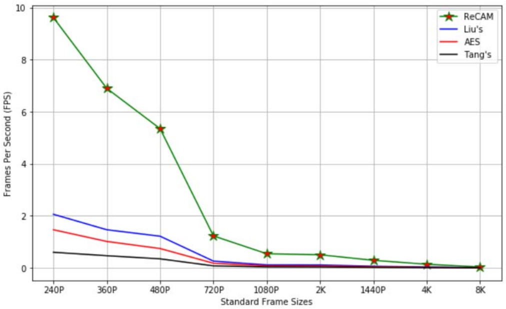

Fig. 10. Comparison of the four schemes in terms of frames per second (fps) for different RGB frame sizes.

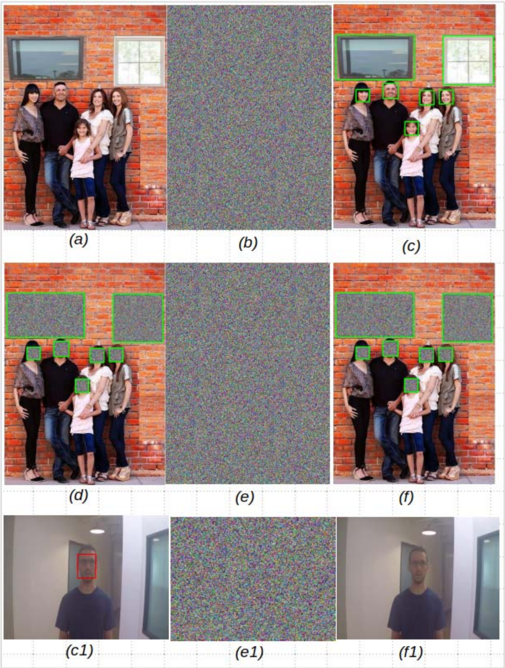

Fig. 11. Window and Face Detection and Denaturing.

Figure 11 shows how a window and face containing frame is processed at the point of creation, at a server, and how it is eventually forwarded to the viewing station.

While it is ideal to conduct the whole frame enciphering and objects detection at the edge, it is not economical due to the resource constraints. It is better to perform the whole frame scrambling at the edge following foreground object detection, and the window-and-face objects detection and denaturing at the server. No content of the frame, including the windows and faces, is disclosed over the communication channels. As shown in Fig. 11, windows and faces are detected at the server, denatured, and whole frame is re-encrypted and forwarded to the viewing station. At the viewing station, the whole frame is decrypted but the windows and faces remain denatured to prevent peeping and ensure anonymity. However, faces of fugitives, as identified by the SVM model, are not denatured. Rather they are forwarded to the SOC as they are (Fig. 11(f1)).

VIII. CONCLUDING REMARKS

In this paper, we proposed PriSE, a solution for privacy-preserving surveillance as an edge service. A novel slender video frame scrambling scheme, ReCAM, is in introduced to ensure end-to-end privacy of frame contents, and a window-and-face object detector is applied to identify and denature windows and faces on frames. The security and computational efficiency of the scrambling technique were tested and analyzed extensively using standard cryptographic analysis techniques and apropos performance metrics. Besides, the window-object detection method was trained and tested using various types of windows. The model loaded on fog/cloud server can detect windows with an accuracy of 99.3% and the MTCNN based face-detector can detect faces with an accuracy of 95.5%. In addition, our experimental results show that the scrambling scheme can process 5.61 fps on a Raspberry PI 4. But when coupled with the simplified motion detection scheme, the overall processing speed at the edge becomes 5.12 fps. Hence, the experimental studies and analyses validated that the PriSE is able to efficiently scramble frames and recognize and denature windows to prevent peeping in real-time. Besides, employing an edge device with more powerful computational resources can drastically enhance the processing speed.

References

[1] M. Al-Rubaie and J. M. Chang, "Privacy-preserving machine learning: Threats and solutions," *IEEE Security & Privacy*, vol. 17, no. 2, pp. 49–58, 2019.

[2] R. Altawy and A. M. Youssef, "Security, privacy, and safety aspects of civilian drones: A survey," *ACM Transactions on Cyber-Physical Systems*, vol. 1, no. 2, p. 7, 2017.

[3] P. Birnstill, *Privacy-Respecting Smart Video Surveillance Based on Usage Control Enforcement*. KIT Scientific Publishing, 2016, vol. 25.

[4] A. Cavailaro, "Privacy in video surveillance [in the spotlight]," *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 168–166, 2007.

[5] A. Cavoukian, *Privacy and drones: Unmanned aerial vehicles*. Information and Privacy Commissioner of Ontario, Canada Ontario, 2012.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[7] A. Fitwi, Y. Chen, and N. Zhou, "An agent-administrator-based security mechanism for distributed sensors and drones for smart grid monitoring," in *Signal Processing, Sensor/Information Fusion, and Target Recognition XXVIII*, vol. 11018. International Society for Optics and Photonics, 2019, p. 110180L.

[8] A. Fitwi, Y. Chen, and S. Zhu, "A lightweight blockchain-based privacy protection for smart surveillance at the edge," *1st International Workshop on Lightweight Blockchain for Edge Intelligence and Security (LightChain, colocated with IEEE BlockChain Conference)*, 2019.

[9] A. Fitwi, M. Yuan, S. Y. Nikouei, and Y. Chen, "Minor privacy protection by real-time children identification and face scrambling at the edge," *EAI Endorsed Transactions on Security and Safety: Online First*, 5 2020.

[10] A. H. Fitwi and S. Nouh, "Performance analysis of chaotic encryption using a shared image as a key," *Zede Journal*, vol. 28, pp. 17–29, 2011.

[11] A. B. Godbehere, A. Matsukawa, and K. Goldberg, "Visual tracking of human visitors under variable-lighting conditions for a responsive audio art installation," in *2012 American Control Conference (ACC)*. IEEE, 2012, pp. 4305–4312.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[15] B. Jiang, Q. Ren, F. Dai, J. Xiong, J. Yang, and G. Gui, "Multi-task cascaded convolutional neural networks for real-time dynamic face recognition method," in *International Conference in Communications, Signal Processing, and Systems*. Springer, 2018, pp. 59–66.

[16] L. Liu and S. Miao, "A new image encryption algorithm based on logistic chaotic map with varying parameter," *SpringerPlus*, vol. 5, no. 1, p. 289, 2016.

[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[18] J. M. Myerson, "Identifying enterprise network vulnerabilities," *International Journal of Network Management*, vol. 12, no. 3, pp. 135–144, 2002.

[19] S. Y. Nikouei, Y. Chen, S. Song, and T. R. Faughnan, "Kerman: A hybrid lightweight tracking algorithm to enable smart surveillance as an edge service," in *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2019, pp. 1–6.

[20] S. Y. Nikouei, R. Xu, D. Nagothu, Y. Chen, A. Aved, and E. Blasch, "Real-time index authentication for event-oriented surveillance video query using blockchain," in *2018 IEEE International Smart Cities Conference (ISC2)*. IEEE, 2018, pp. 1–8.

[21] Y. Pang, Y. Yuan, X. Li, and J. Pan, "Efficient hog human detection," *Signal Processing*, vol. 91, no. 4, pp. 773–781, 2011.

[22] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992.

[23] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Booz-Allen and Hamilton Inc Mclean Va, Tech. Rep., 2001.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[25] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[26] seeprettyface, "http ://www.seeprettyface.com/mydataset.html, last accessed on may 05, 2020," 2018.

[27] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y.-L. Tian, A. Ekin, J. Connell, C. F. Shu, and M. Lu, "Enabling video privacy through computer vision," *IEEE Security & Privacy*, vol. 3, no. 3, pp. 50–57, 2005.

[28] A. Sharma and H. Foroosh, "Slim-cnn: A light-weight cnn for face attribute prediction," *arXiv preprint arXiv:1907.02157*, 2019.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[30] C. Slobogin, "Peeping techno-toms and the fourth amendment: Seeing through kyllo's rules governing technological surveillance," *Minn. L. Rev.*, vol. 86, p. 1393, 2001.

[31] C. Streiffer, A. Srivastava, V. Orlikowski, Y. Velasco, V. Martin, N. Raval, A. Machanavajjhala, and L. P. Cox, "eprivateeye: To the edge and beyond!" in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. ACM, 2017, p. 18.

[32] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[33] Z. Tang, Y. Yang, S. Xu, C. Yu, and X. Zhang, "Image encryption with double spiral scans and chaotic maps," *Security and Communication Networks*, vol. 2019, 2019.

[34] J. Thornton, J. Baran-Gale, D. Butler, M. Chan, and H. Zwahlen, "Person attribute search for large-area video surveillance," in *2011 IEEE International Conference on Technologies for Homeland Security (HST)*. IEEE, 2011, pp. 55–61.

[35] E. Vattapparamban, İ. Güvenç, A. İ. Yurekli, K. Akkaya, and S. Uluağaç, "Drones for smart cities: Issues in cybersecurity, privacy, and public safety," in *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2016, pp. 216–221.

[36] P. Viola, M. Jones *et al.*, "Rapid object detection using a boosted cascade of simple features," *CVPR (1)*, vol. 1, no. 511-518, p. 3, 2001.

[37] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan, "Enabling live video analytics with a scalable and privacy-aware framework," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 3s, p. 64, 2018.

[38] R. Wu, Y. Chen, E. Blasch, B. Liu, G. Chen, and D. Shen, "A container-based elastic cloud architecture for real-time full-motion video (fmv) target tracking," in *2014 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. IEEE, 2014, pp. 1–8.

[39] R. Wu, B. Liu, Y. Chen, E. Blasch, H. Ling, and G. Chen, "A container-based elastic cloud architecture for pseudo real-time exploitation of wide area motion imagery (wami) stream," *Journal of Signal Processing Systems*, vol. 88, no. 2, pp. 219–231, 2017.

[40] J. Yu, B. Zhang, Z. Kuang, D. Lin, and J. Fan, "iprivacy: image privacy protection by identifying sensitive objects via deep multi-task learning," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1005–1016, 2017.

[41] M. Yuan, S. Y. Nikouei, A. Fitwi, Y. Chen, and Y. Dong, "Minor privacy protection through real-time video processing at the edge," *arXiv preprint arXiv:2005.01178*, 2020.

[42] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[43] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.