Hindawi
Security and Communication Networks
Volume 2018, Article ID 3980371, 10 pages
https://doi.org/10.1155/2018/3980371

Research Article

User Presence Inference via Encrypted Traffic of Wireless Camera in Smart Homes

Xiaoyu Ji, Yushi Cheng, Wenyuan Xu, and Xinyan Zhou

Ubiquitous System Security Lab (USSLAB), Zhejiang University, Hangzhou 310027, China

Correspondence should be addressed to Wenyuan Xu; wyxu@zju.edu.cn

Received 15 June 2018; Accepted 6 September 2018; Published 25 September 2018

Guest Editor: Wei Li

Wireless cameras are widely deployed in smart homes for security guarding, baby monitoring, fall detection, and so on. Those security cameras, which are supposed to protect users, however, may in turn leak a user's personal privacy. In this paper, we reveal that attackers are able to infer whether users are at home or not, that is, the user presence, by eavesdropping the traffic of wireless cameras from distance. We propose `HomeSpy`, a system that infers user presence by inspecting the intrinsic pattern of the wireless camera traffic. To infer the user presence, `HomeSpy` first eavesdrops the wireless traffic around the target house and detects the existence of wireless cameras with a Long Short-Term Memory (LSTM) network. Then, `HomeSpy` infers the user presence using the bitrate variation of the wireless camera traffic based on a cumulative sum control chart (CUSUM) algorithm. We implement `HomeSpy` on the Android platform and validate it on 20 cameras. The evaluation results show that `HomeSpy` can achieve a successful attack rate of 97.2%.

1. Introduction

Home security cameras are widely deployed in smart homes to provide protection services ranging from home security through baby monitoring to fall detection. Wi-Fi wireless cameras, that is, Internet protocol cameras equipped with Wi-Fi modules, are booming among home security camera market due to their flexibility and usability. According to Technavio [1], the global wireless video surveillance market will continue to grow at a rate of 21.35% during the period 2014–2019. However, despite its popularity and convenience, we discover that wireless cameras, which are designed to provide protection services, are likely to leak personal privacy and become a point of attacking.

Already, an increasing attention has been paid to the privacy issues caused by wireless cameras [2–4]. Much work has been proposed to safeguard personal privacy against cameras. This work [5] introduces an "invisible light beacon" implemented on the eye-wear to prevent unauthorized video-taping, by which the privacy preferences of photographed users are communicated to photographing cameras. These authors [6–9] focus on the privacy concerns caused by "first-person" wearable cameras. They propose methods to identify and prevent the sharing of sensitive images captured by wearable cameras. Birnbach et al. [10] detect drones carrying out privacy invasion attacks with on-board cameras. This work analyzes the RSSI (received signal strength indicator) of the wireless traffic from the camera on the drone to detect its approach.

Different from previous work, our paper focuses on the wireless security cameras and reveals that wireless cameras may leak user privacy, that is, the user presence. We propose `HomeSpy`, a system that is able to infer whether users are at home or not, by eavesdropping the wireless camera traffic. As shown in Figure 1, an adversary tries to figure out whether there are any users inside a target house. She runs `HomeSpy` on her smartphone outside the house to infer the user presence. `HomeSpy` overhears and analyzes the wireless traffic, even it is encrypted, and informs the adversary that the owner is likely not at home. With this information, the adversary can conduct further malfeasance, that is, wade the house physically.

Inferring user presence via wireless camera traffic is promising yet challenging. Since an adversary has no access to the house as well as its Wi-Fi network, `HomeSpy` shall work without joining the network. In addition, there might

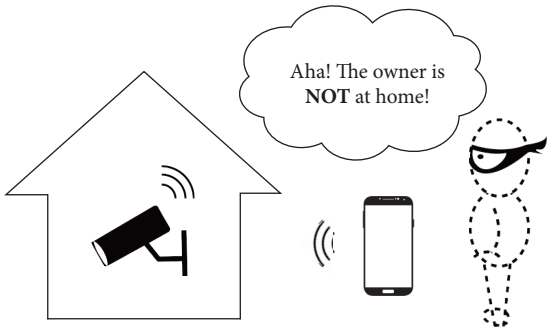

Figure 1: HomeSpy is able to infer the user presence by eavesdropping the wireless camera traffic.

be various devices inside the house that generate wireless traffic; HomeSpy shall figure out whether there is a wireless camera and which packets belong to the wireless camera. Furthermore, camera traffic is encrypted and thus traditional image/audio processing techniques are invalid in this scenario. HomeSpy shall exploit a new angle to reveal the information ensconced in the camera traffic.

Our key insight is that compression and fragmentation over video/audio streams have fundamental impact on the traffic of wireless cameras, which appears as a sequence of full-size packets with small-size packets in between. Such a traffic pattern shall make the wireless camera distinguishable from other network applications. In addition, the number and size of video/audio frames of wireless camera traffic are not fixed and depend on the video/audio content. If a user is within the filming range of the wireless camera, the frame number and size are likely to change with the human intervention (i.e., motion and sound). By exploring the human intervention in the wireless camera traffic, HomeSpy shall be able to infer whether the user is at home or not, that is, the user presence.

To overcome all the aforementioned challenges, HomeSpy eavesdrops the wireless traffic near the target house with a smartphone and detects the existence of wireless cameras inside the house with a LSTM network. Then, HomeSpy inspects the bitrate variation of the camera traffic and infers the user presence with a CUSUM algorithm. In summary, our contribution includes the following.

(i) We reveal that wireless surveillance cameras in smart homes are potential sources of privacy leakage, that is, the user presence status, which can result in serious security issues.

(ii) We propose HomeSpy, a system that is able to analyze whether owners are at home or not, by eavesdropping the network traffic of wireless security cameras.

(iii) We implement HomeSpy on the Android platform and validate it on 20 popular wireless security cameras. The results demonstrate that HomeSpy can successfully attack with a probability of 97.2%.

The rest of the paper proceeds as follows. First, we outline the background and insights of HomeSpy in Section 2. Then, we present the threat model of HomeSpy in Section 3. In Section 4, we elaborate the design of HomeSpy in detail, and in Section 5, we present the implementation details and experimental results. Discussion and related work are provided in Sections 6 and 7, respectively, with Section 8 giving concluding remarks.

2. Background

In this section, we first outline the basics of the wireless camera and, then, its intrinsic traffic patterns exploited by HomeSpy.

2.1. Basics of Wireless Camera. Wireless security cameras provide real-time video surveillance through a wireless network operating at license-free frequencies (e.g., 2.4 GHz). Instead of saving records locally, wireless cameras process the video/audio streams and then upload them through an access point (AP) of a wireless local area network (WLAN) to a remote cloud server. Modern wireless cameras, therefore, allow users to achieve remote monitoring by accessing the cloud server.

The hardware modules of a wireless camera are shown in Figure 2. Wireless cameras monitor target environments, for example, homes and offices, with build-in CMOS/CCD sensors, which capture and digitize the image scenes to generate raw video and audio frames. The raw multimedia frames are then fed into an SOC (system on chip) before being transmitted to a remote client or server by the Wi-Fi chip. A multimedia SOC chip usually contains a MCU and three additional submodules: (1) image signal processing (ISP) submodule that performs functions such as noise filtering; (2) codec submodule that compresses video/audio frames to decrease frame sizes; (3) networking protocol submodule that transmits multimedia stream via streaming protocols like RTSP (real-time streaming protocol).

2.2. Wireless Camera Traffic Pattern. As mentioned above, the codec submodule compresses video/audio frames to decrease frame sizes. Various brands of cameras may have slightly different implementation, but most of them employ popular compression standards, for example, H.264 [11] for video and AAC [12] for audio. H.264 standard utilizes the redundancy between consecutive frames and outputs a series of Intra coded frames (I-frame), Predicted frames (P-frame), and Bi-directional predicted frames (B-frame), for example, a sequence of $\{I, P, B, P, B, P, B, I, \ldots\}$ [13], as shown in Figure 3. I frames are coded without reference and can be decoded by themselves. P frames are coded based on the prior frames, while B frames are coded based on both the prior and the subsequent frames. The number of P and B frames between two consecutive I frames is not fixed and depends on the video content. Since I frames do not utilize inter-frame redundancy, they typically are a few times larger than the other two frame types, and P frames are larger than B frames. An audio stream is compressed similarly but with a much lower bandwidth. Audio signals are sampled continuously, and encoding protocols (e.g., AAC) are used on the audio samples to remove redundancy.

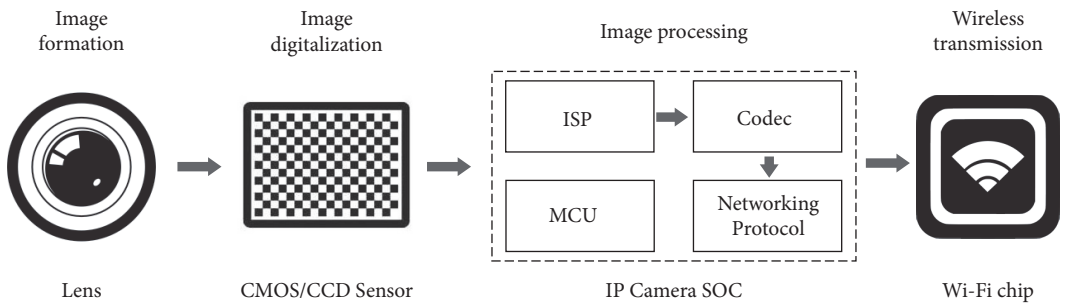

Figure 2: Hardware modules inside a wireless camera.

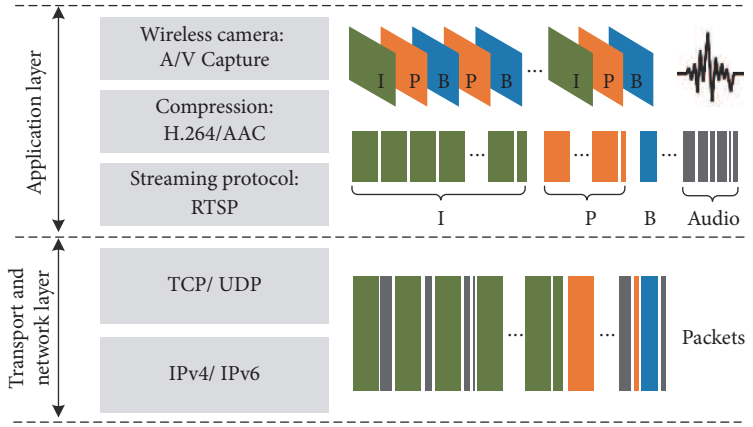

Figure 3: Compression and fragmentation of wireless camera stream.

After being compressed, the video and audio stream frames go through the application layer to be encapsulated with networking protocols, such as RTSP, before being transmitted in the data link layer. Since the network interface in the link layer can only transmit a packet of a size smaller than its maximum transmission unit (MTU), when a frame is larger than MTU, it has to be fragmented. For video streams, since I frames are typically the largest among the three types, one I frame is divided into the largest number of packets. Denote the number n, and then the first $n - 1$ packets are *full-size packets* (i.e., MTU) and the last one may be smaller than the full size, which we call *small-size packet*. B frames are typically the smallest, and they may fit in one packet or a few packets. In comparison, the frames of audio streams are smaller than videos and thus, a less number of full-size packets are formed. In general, video and audio frames are divided into both full-size packets and small-size packets as follows:

(i) I frames are divided into a set of full-size packets followed by a small-size packet.

(ii) P frames are divided into a smaller set of full-size packets followed by a small-size packet.

(iii) Most B frames and audio frames contribute to small-size packets.

As such, compression and fragmentation over video/audio streams form a unique traffic pattern, which can be exploited to differentiate wireless cameras from other network applications.

2.3. Human Impact on Wireless Camera. For the sake of privacy protection, the camera traffic is encrypted in the SOC chip. Therefore, an adversary cannot obtain the video/audio content from the traffic directly. However, after conducting an in-depth study on the working principle of wireless cameras, we find that even if encrypted, the camera traffic can still reveal privacy information.

As mentioned above, the number and size of video frames are not fixed and depend on the video content. When the video content changes, the frame number and size increase to capture the discrepancy of the content. Therefore, if a user is within the filming range of the wireless camera, the number and size of the frames are likely to change with the human motion. Similarly, the voice from the user also increases the audio traffic. Thus, both motions and voices from a user can increase the frame number and size of the wireless camera, resulting in a larger amount of traffic and thus a higher bitrate. This finding sheds light upon the insights of `HomeSpy`. As humans may impact the traffic of the inside wireless camera, the bitrate of the wireless camera may in turn reveal whether there is any person inside the house.

3. Threat Model

In this section, we present the threat model of `HomeSpy`. Since the adversary's goal is to infer user presence from distance, we consider the following attack scenario: *in a residential area, an adversary attempts to commit crimes, such as burglary. To*

reduce the chance of getting caught, the adversary tends to choose houses without the presence of owners. To achieve it, the adversary utilizes HomeSpy *to infer the user presence status*. In such a scenario, we assume the adversary has following abilities.

Vicinity to Target House. We assume that the adversary may target any houses of their choices and can be in the target house's vicinity and draw no attention.

No Prior Information of Wireless Camera. Although the adversary may obtain some prior knowledge of the target house, to increase the attack difficulty, we assume they have no information whether there is any wireless camera inside the house.

No Access to Wi-Fi Network. We assume the adversary has no password for the Wi-Fi network in the target house. Thus, they have no access to the Wi-Fi network and cannot collect the wireless traffic by joining the network.

No Specific User Interaction. The adversary is unable to ask users to perform any operations. Thus, they cannot rely on any specific user operations, for example, making a phone call or running a specific application, but natural user behaviors that are introduced frequently at home.

4. Attack Design

Based on the threat model mentioned above, we design HomeSpy to enable the adversary to infer user presence from distance. In this section, we elaborate the attack design in detail.

4.1. Workflow of HomeSpy. The workflow of HomeSpy is composed of four steps: traffic eavesdropping, flow construction, camera detection, and bitrate analysis. First, HomeSpy sets the Wi-Fi card on the smartphone into the monitor mode to capture network traffic over wireless channels near the target house. Then, HomeSpy filters non-data and downlink packets and constructs the remaining packets into flows based on their MAC addresses. In the camera detection step, HomeSpy feeds the packet length sequence of each flow into a LSTM network and detects the existence of wireless cameras. Finally, HomeSpy examines the bitrate variation of the detected camera flow with a CUSUM algorithm to infer the user presence.

4.2. Traffic Eavesdropping. A straightforward method to overhear the network traffic is to join the network; however, an adversary usually has no access to the Wi-Fi password; even if they do, most WLANs (wireless local area networks) are encrypted with WPA/WPA2-PSK [14]. These methods exploit per-client, per-session keys, which are derived from the Wi-Fi password and the information exchanged when a client joins the network [15]. As a result, even with the Wi-Fi password, it is difficult for the adversary to capture all four handshake packets to derive the keys.

To eavesdrop the wireless traffic of the target house without joining the network, HomeSpy collects data by setting the smartphone's Wi-Fi card into the monitor mode. Normally, a smartphone sets its Wi-Fi card in the managed mode, in which packets not destined for this smartphone are discarded. However, HomeSpy requires to eavesdrop all the nearby wireless traffic for analysis. To this end, the Wi-Fi card shall be set into the monitor mode such that HomeSpy is able to capture and record all the passing wireless packets.

We implement the monitor mode function on the Android platform based on an open-source project named Nexmon [16]. Nexmon provides a basic API for Wi-Fi driver modification. We use a UDP socket to read packets from the `rawproxy` application, which connects to the Wi-Fi card buffer in Nexmon. Once receiving a packet via the UDP socket, the packet is decoupled and collected for further analysis. In this way, HomeSpy is able to eavesdrop the network traffic on the sly.

4.3. Flow Construction. In the flow construction step, HomeSpy first filters packets that are unlikely to be from camera applications. Since wireless cameras mainly create uploading streams with a destination of an access point, we remove both non-data packets and downlink packets. Non-data packets include ACK, RTS, CTS, and other management packets, and downlink flows are identified by reading the `FC` field in the MAC frame header. HomeSpy only records the `length` and `address` fields of each packet and discards the payload, for the sake of efficiency.

Then, HomeSpy groups the remaining packets into flows according to their MAC addresses. In this step, we put packets with identical source and destination MAC addresses into the same group and regard all packets belonging to the same MAC pair as an individual flow.

4.4. Camera Detection. As revealed in Section 2, compression and fragmentation over video/audio streams have fundamental impact on the wireless camera traffic pattern. The H.264 and AAC standards indicate that when video/audio streams are passed to the network interface cards, a sequence of I, P, B, and audio frames are packaged into a series of full-size packets with small-size packets in between.

HomeSpy detects the wireless camera flow by exploiting this packet length pattern. However, since different cameras have various bitrates and resolutions, and the packet length sequences of different applications can diverse significantly in terms of length even within the same time period, it is difficult to extract robust features manually. We build a Long Short-Term Memory network [17] that excels at dealing with time sequences and can learn discriminative features from samples by itself. The built LSTM network takes the packet length sequence of each flow as the input and outputs whether it is a wireless camera flow. This completely data driven approach, compared to traditional hand crafted feature extraction methods, leads to much simpler-to-use and more reliable outcomes in practice.

Moreover, to lighten the network overhead and achieve real-time attacks, we implement the LSTM network on the smartphone instead of uploading raw data to the cloud and running classification algorithms remotely. Due to the limited resources and computational capability on the mobile

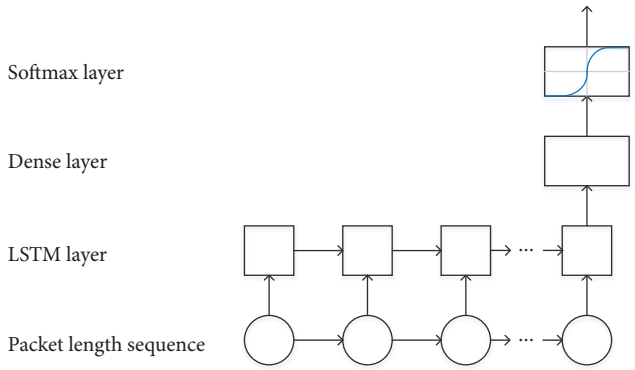

Figure 4: Network architecture of the proposed LSTM model.

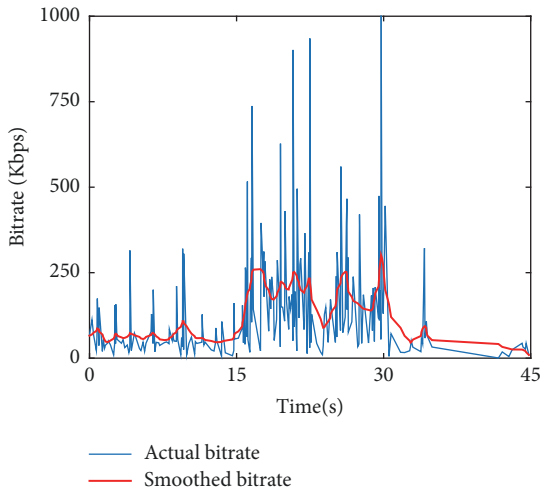

Figure 5: Impact of human beings on the bitrate of the camera flow.

devices, the LSTM network employed on the smartphone shall be lightweight. The architecture of the employed LSTM network is shown in Figure 4, where we only use one LSTM layer for the sake of lightweight. In addition, we elaborately select the parameters of the network in Section 5, that is, the number of neurons in the LSTM layer $m = 16$, the number of neurons in the dense layer $n = 16$, the times of iteration $k = 5$, and the size of batching $b = 64$, to strike the balance of accuracy and efficiency on the smartphone.

Note that this wireless camera detection approach targets nonintensive houses. For intensive residences, for example, apartments, we can further infer in which room the wireless camera is installed by widely studied wireless localization techniques [18–22].

4.5. Bitrate Analysis. As mentioned in Section 2, if a person is within the filming range of the wireless camera, the frame number and size will change with the human intervention, as well as the bitrate. An illustration is shown in Figure 5, in which a human keeps moving during the period from 15 *s* to 30 *s*. As a result, the bitrate of the camera flow also shows a rise and fall during this period with only a one-second lag. With this observation, we employ the bitrate variation of the detected camera flow to infer the user presence status: present or absent.

To detect the changes of bitrate caused by human beings, HomeSpy utilizes the cumulative sum control chart (CUSUM) [23] algorithm to detect the rising edges of the bitrate sequence r. CUSUM is a sequential analysis technique typically used for change monitoring. The CUSUM algorithm for bitrate variation detection is as follows:

$$U_n = \begin{cases} 0, & n = 0 \\ \max\left(0, U_{n-1} + r_n - w_n\right), & n > 0 \end{cases} \quad (1)$$

$$Condition : U_n > \delta, \quad n = 0, 1 \ldots N \quad (2)$$

where U_n is the upper cumulative sum at the time n. w is the likelihood estimation of the bitrate sequence. δ is the threshold for detecting inside human beings. If the value of U_n exceeds δ, HomeSpy outputs that the owners of the house are at home; otherwise, HomeSpy believes they may have gone out.

5. Performance Evaluation

In this section, we evaluate the performance of HomeSpy. We first present the setup of the experiments, then the evaluation metrics, and finally the results of HomeSpy attack.

5.1. Experimental Setup

Experiment Scene. We perform experiments in a house with a wireless camera installed in the bedroom. The adversary is outside the house with a distance of 3 *m* and runs the HomeSpy application on an LG Nexus 5 smartphone in real-time to infer the user presence. Note that the attack distance is set as 3 *m* for illustration but not limited to it. In fact, HomeSpy is effective as long as the attacker can capture the wireless traffic of the home security camera. Therefore, the effective attack distance of HomeSpy roughly equals to the Wi-Fi covering range.

Camera Brands. We select 20 typical cameras of 12 well-known manufacturers on the market from both US and China in our experiments, and their detailed information including brand, model, compression protocol, and default resolution is summarized in Table 1.

5.2. Performance Metrics. We use *precision*, *recall*, and *F1-score* to evaluate the performance of HomeSpy on wireless camera detection and utilize *true positive rate (TPR)*, *true negative rate (TNR)*, and *successful attack rate (SAR)* to evaluate the performance of HomeSpy on user presence inference.

Precision. We define precision as

$$precision = \frac{TP}{FP + TP} \quad (3)$$

where TP and FP represent the true positives and the false positives of wireless camera detection, respectively.

Table 1: Summary of experimental cameras.

No.	Brand	Model	Compression	Default Resolution
1	Dahua	LeChange TC1	H.264, AAC	720p
2	Dahua	LeChange TC5	H.265, AAC	1080p
3	Dahua	LeChange TC6C	H.264, AAC	720p
4	Dahua	LeChange TC7C	H.264, AAC	720p
5	Hikvision	Ezviz C2C	H.264, AAC	720p
6	Hikvision	Ezviz C2mini	H.264, AAC	720p
7	Yi	1	H.264, AAC	720p
8	Yi	2	H.264/MJPEG, AAC	1080p
9	Xiaomi	iSC5	H.264, AAC	1080p
10	Xiaomi	SXJ01ZM	H.264, AAC	1080p
11	360	D600	H.264, Opus	720p
12	360	D606	H.264, Opus	1080p
13	TP-LINK	TL-IPC20-2.8	H.264, -	720p
14	TP-LINK	TL-IPC10A	H.264, AAC	720p
15	Nest	Cam Indoor	H.264, AAC	1080p
16	Amcrest	IP2M-841W	H.264, AAC	1080p
17	D-Link	DCS-820L	H.264/JPEG, AAC	1080p
18	Lenovo	G2 Mini	H.264/JPEG, AAC	720p
19	Haier	HC6700	H.265/JPEG, AAC	720p
20	Yoosee	KP-01	H.264, AAC	720p

Recall. We define recall as

$$recall = \frac{TP}{FN + TP} \tag{4}$$

where FN is the false negatives of wireless camera detection.

F1-Score. We define F1-score as

$$F1 - score = \frac{2 \times (precision * recall)}{precision + recall} \tag{5}$$

to strike the balance of precision and recall.

Successful Attack Rate. We define TPR and TNR as the rate of correctly inferring user presence and absence, respectively. Since HomeSpy attacks the wireless camera to infer the user presence, both true positive and true negative are successful attacks. Hereby, we define

$$SAR = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{6}$$

as the successful attack rate.

5.3. Wireless Camera Detection Performance. In the first set of experiments, we evaluate the wireless camera detection performance of HomeSpy with aforementioned 20 cameras. For each camera, we collect 100 traces (each lasting 10 s) when the camera working inside the house with 7 other network devices nearby (e.g., mobile phones, laptops, and tablets), which also generate wireless traffic. Note that we regard each camera flow as a positive sample and each non-camera flow as a negative sample. Thus, a collected trace contains a positive sample and several negative samples.

We randomly choose 10% of the samples as the training set, and the remaining samples serve as the testing set. To adapt to the source limited environment on smartphones, we evaluate the appropriate size of m (the number of neurons in the LSTM layer), n (the number of neurons in the dense layer), k (the times of iteration), and b (the size of batching). The results are demonstrated in Figures 6–8, from which we have following findings. First, with the increase in m and n, the performance of HomeSpy is slightly enhanced at a cost of a larger network and more calculation. We set m and n both to be 16 to obtain a small but efficient network on smartphones. Second, with the increase in k, the performance of HomeSpy is slightly improved at a cost of a larger computation delay, while a larger batching size b can reduce the delay at a cost of performance decrease. To balance the computation delay and accuracy, we set the $k = 5$ and $b = 64$. With the selected parameters, HomeSpy can achieve an overall wireless camera detection performance of 98.1% precision, 98.3% recall, and 98.2% F1-score across 20 cameras on smartphones.

5.4. User Presence Inference Performance. In the second set of experiments, we evaluate the user presence inference performance of HomeSpy with 3 cameras (Ezviz, Dahua, and Yi). Three volunteers participated in the experiments. To simulate the present status, the volunteers are asked to walk around for 15 seconds inside the house. For the absent status, the house has no human beings inside. For each status, we utilize HomeSpy for inference and repeat this process for 30 times.

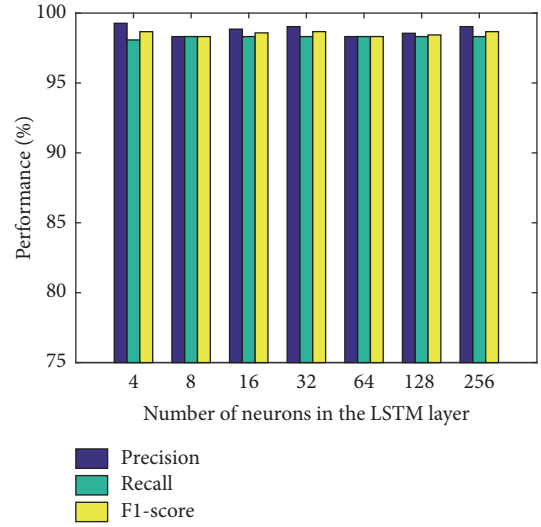

Figure 6: Impact of the number of neurons in the LSTM layer m (with $n = 32$, $k = 10$, and $b = 32$).

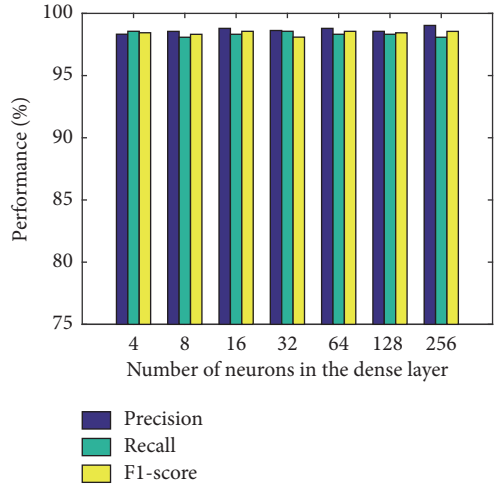

Figure 7: Impact of the number of neurons in the dense layer n (with $m = 16$, $k = 10$, and $b = 32$).

The results in Figure 9 reveal that HomeSpy can successfully attack with an average SAR of 97.2%. False positives mainly come from the fluctuation of bitrate resulting from the dynamic of network environment. Based on the ROC (receiver operating characteristic) curve in Figure 10, we utilize a large threshold and sacrifice a bit of accuracy to achieve a low false positive rate.

Next, we consider several factors that could affect our ability to infer the user presence.

5.4.1. Impact of Motion Range. Since the bitrate variation has resulted from human motions, the motion range may affect the inference performance. A larger motion range can result

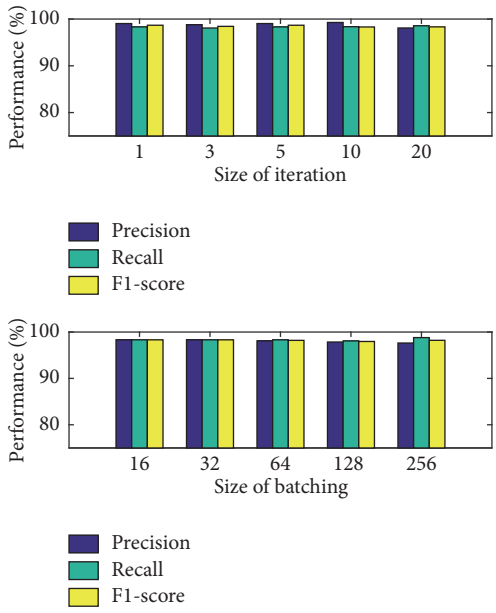

Figure 8: Impact of the iteration times k (with $m = 16$, $n = 16$, and $b = 32$) and the batching size b (with $m = 16$, $n = 16$, and $k = 5$).

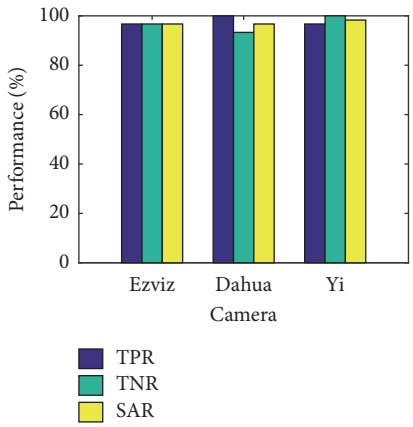

Figure 9: Overall user presence inference performance of HomeSpy.

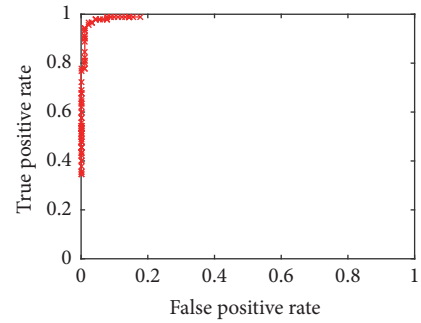

Figure 10: The ROC curve of user presence inference.

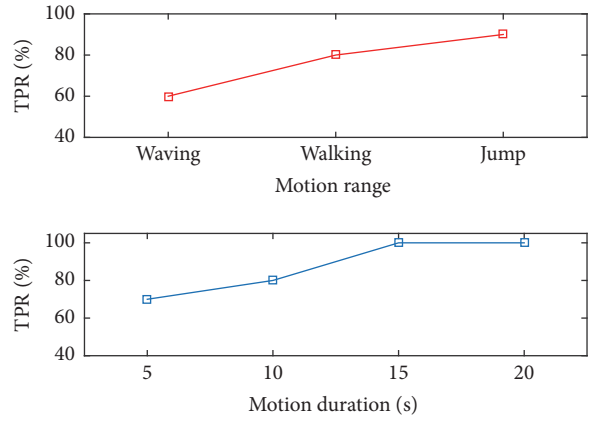

Figure 11: The impact of motion range and motion duration.

in more bitrate variation and, thus, may improve the inference accuracy.

To investigate the impact of motion range, we conduct experiments with three common motions: waving (wave hands), walking (walk around the room), and jump (jump up and down). Apparently, the intensity of motion is strengthened from front to back. The motion duration is set to 10 *s* and 10 attacks are launched for each motion. The results in Figure 11 show that the TPR does increase with the growth of motion range. Even with slight movements such as waving hands, HomeSpy can still successfully attack with a rate of 60%, while jump holds the highest probability of 90%.

5.4.2. Impact of Motion Duration. Another influencing factor of HomeSpy is the motion duration time. Long duration of human intervention helps overcome the transient variation of bitrate caused by the dynamic network environment and, thus, may decrease the false positive rate.

To evaluate the impact of motion duration, we conduct experiments with four typical time durations: 5 *s*, 10 *s*, 15 *s* and 20 *s*. The results are also revealed in Figure 11, which confirm that the longer the time duration is, the better the attack accuracy will be. Even with only 5 *s* of human intervention, HomeSpy is able to successfully attack with a rate of 70%. With the increase in the duration time, the TPR is promoted to 100% for 15 *s* as well as 20 *s*.

5.4.3. Impact of Household Pet. In real life, many families have pet dogs or cats. It is likely that household pets introduce motion or sound as well and, thus, cause the variation of wireless camera traffic. To investigate the impact of pets on the user presence inference, we perform HomeSpy attack with a medium-size household dog. For the present status, both the volunteer and the dog are inside the house, whereas only the dog stays in the room. For each status, we utilize HomeSpy for inference and repeat this process 30 times.

The results in Figure 12 reveal that the pet dog does not impact the true positive rate of HomeSpy (97.8% on average). However, false positives rise due to the impact of household pets, which decrease the true negative rate. Nevertheless, even with the interference from a household pet, HomeSpy can successfully attack with an average SAR of 91.1%. In addition, we argue that as the goal of the attacker is to commit crimes,

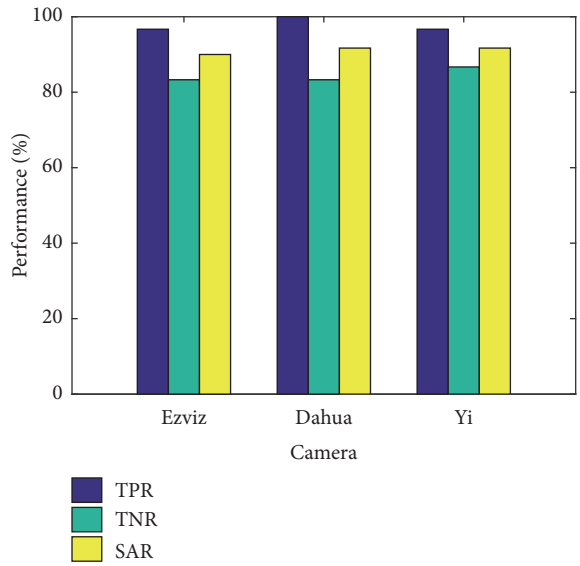

Figure 12: The impact of pets on the user presence inference.

for example, burglary, they shall try to avoid any attention, even that of pets. A break-in may cause a household dog to bark and thus alarm the neighbors. Therefore, from the perspective of the attacker, they shall consider the case of pets and choose a small threshold to achieve a low false negative rate to reduce the chance of being caught.

6. Discussion

With the elaborate design, HomeSpy achieves high performance most of the time. However, HomeSpy has the following limitations.

First, HomeSpy is mainly applicable to houses with wireless cameras for home monitoring. For houses with no cameras, or with local-storage cameras or wired cameras, HomeSpy is inapplicable. However, considering the booming market of wireless cameras, we believe HomeSpy can attack a large number of houses and infer their user presence status.

Second, HomeSpy targets nonintensive houses. For intensive residences, for example, apartments, HomeSpy needs to further determine whether the detected wireless camera is inside the target apartment or not. However, we assume it can be achieved by widely studied wireless localization techniques [18–22] and keep it as the further work.

Third, HomeSpy is designed based on the general principle of wireless cameras in today's market, that is, compression and fragmentation on video and audio streams. If the camera working mode changes dramatically in the future, HomeSpy may not function well and should be updated to incorporate the new working mode accordingly.

Fourth, HomeSpy utilizes the human intervention (i.e., motion and sound) reflected on the camera traffic to infer the user's presence. Thus, HomeSpy works in the scenario where limited motion is performed, but sounds exist, for example, watching movies. However, if the user is not within the filming range of the camera or displays both limited motion and sound (e.g., sleeping or reading), they may not cause enough variation on the camera bitrate and, thus, render HomeSpy ineffective.

7. Related Work

Privacy Concerns with Cameras. Much attention has been paid to the privacy issues brought by the widespread cameras. Existing literature safeguards personal privacy against cameras from several aspects. The first type detects hidden cameras to ensure personal privacy. This work [10] detects drones that carry out privacy invasion attacks with on-board cameras by analyzing RSSI signals. A few applications available on both the Android and iOS platforms detect hidden cameras against secret videotaping. They utilize the signs of light reflection caused by the lenses of a hidden wireless camera [24–26], or the electromagnetic waves (produced by the crystal oscillator) emitted by working cameras [27, 28] to detect hidden cameras. This work [9], on the other hand, prevents unauthorized videotaping by introducing an "invisible light beacon" implemented on the eye-wear. Another type of work [5–8] focuses on the privacy concerns caused by "first-person" wearable cameras and proposes methods to identify and prevent sharing of sensitive images captured by wearable cameras.

Similar to [29], HomeSpy is inspired by previous work and reveals the perspective of attackers that the home security camera can be a potential source of privacy leakage, that is, the user presence status.

Traffic Identification. Essentially, HomeSpy is one type of the traffic identification (TI) problem. However, HomeSpy differs from the traditional TI and is more challenging. Traditional TI approaches [30–34] mainly take the 5-tuple information as inputs, while HomeSpy can only use the PHY/MAC layer information without decoding the TCP/IP headers. This makes HomeSpy challenging and applicable to larger types of scenarios.

8. Conclusion

In this paper, we reveal that wireless cameras in smart homes can leak personal privacy that may result in security issues. We propose HomeSpy, a system that is able to infer whether owners are at home or not, by eavesdropping the traffic of wireless surveillance cameras. We implement HomeSpy on the Android platform and validate it on 20 cameras. The evaluation results show that HomeSpy can achieve a successful attack rate of 97.2%.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work has been funded in part by NSFC 61472358, NSFC 61702451, and the Fundamental Research Funds for the Central Universities 2017QNA4017.

References

[1] Technavio, "Global video surveillance market 2016-2020," http://www.technavio.com/report/global-it-security-video-surveillance-market, 2015.

[2] R. Kenner, "Man sues after finding hidden cam in hotel bathroom," http://www.rense.com/general29/msn.htm, 2012.

[3] A. Castelan and J. Treanor, "Hidden cameras found inside a las vegas airbnb rental recording naked people," https://tinyurl.com/zrtysgx, 2016.

[4] KpopJoA, "Korean couple discover a hidden camera inside airbnb guesthouse in japan," https://tinyurl.com/y9ju55ut, 2017.

[5] A. Ashok, V. Nguyen, M. Gruteser, N. Mandayam, W. Yuan, and K. Dana, "Do not share! Invisible light beacons for signaling preferences to privacy-respecting cameras," in *Proceedings of the 1st ACM MobiCom Workshop on Visible Light Communication Systems, VLCS 2014*, pp. 39–44, USA, September 2014.

[6] R. Templeman, M. Korayem, D. Crandall, and A. Kapadia, "PlaceAvoider: steering first-person cameras away from sensitive spaces," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, 2014.

[7] R. Hoyle, R. Templeman, S. Armes, D. Anthony, D. Crandall, and A. Kapadia, "Pprivacy behaviors of lifeloggers using wearable cameras," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2014*, pp. 571–582, USA, September 2014.

[8] R. Hoyle, R. Templeman, D. Anthony, D. Crandall, and A. Kapadia, "Sensitive lifelogs," in *Proceedings of the the 33rd Annual ACM Conference*, pp. 1645–1648, Seoul, Republic of Korea, April 2015.

[9] M. Korayem, R. Templeman, D. Chen, D. Crandall, and A. Kapadia, "Enhancing lifelogging privacy by detecting screens," in *Proceedings of the the 2016 CHI Conference*, pp. 4309–4314, Santa Clara, Calif, USA, May 2016.

[10] S. Birnbach, R. Baker, and I. Martinovic, "Wi-Fly?: detecting privacy invasion attacks by consumer drones," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, 2017.

[11] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, 2007.

[12] M. Bosi, K. Brandenburg, S. Quackenbush et al., "ISO/IEC MPEG-2 advanced audio coding," *Journal of the Audio Engineering Society*, vol. 45, no. 10, pp. 789–814, 1997.

[13] M. Jiang, X. Yi, and N. Ling, "Improved frame-layer rate control for H.264 using MAD ratio," in *Proceedings of the 2004 IEEE International Symposium on Circuits and Systems*, pp. 813–816, Vancouver, BC, Canada.

[14] W. Group, "Amendment 6: Medium access control (mac) security enhancements," *IEEE Standard Association*, vol. 802, 2004.

[15] A. H. Lashkari, M. M. S. Danesh, and B. Samadi, "A survey on wireless security protocols (WEP, WPA and WPA2/802.11i)," in *Proceedings of the 2nd IEEE International Conference on Computer Science and Information Technology (ICCSIT '09)*, pp. 48–52, IEEE, Beijing, China, August 2009.

[16] M. Schulz, D. Wegemer, and M. Hollick, "Nexmon: The c-based firmware patching framework," https://nexmon.org, 2017.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 37, no. 6, pp. 1067–1080, 2007.

[19] I. Guvenc and C.-C. Chong, "A survey on toa based wireless localization and nlos mitigation techniques," *IEEE Communications Surveys Tutorials*, vol. 11, no. 3, 2009.

[20] R. W. Ouyang, A. K.-S. Wong, and C.-T. Lea, "Received signal strength-based wireless localization via semidefinite programming: noncooperative and cooperative schemes," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 3, pp. 1307–1318, 2010.

[21] Z. Farid, R. Nordin, and M. Ismail, "Recent advances in wireless indoor localization techniques and system," *Journal of Computer Networks and Communications*, vol. 2013, Article ID 185138, 12 pages, 2013.

[22] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu, "STPP: spatial-temporal phase profiling-based method for relative RFID tag localization," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 596–609, 2017.

[23] G. Miller, "Cusum control charts," 1993.

[24] A. L. Poretz, "Spy hidden camera detector," https://tinyurl.com/ybkep49w, 2017.

[25] L. LSC, "Hidden camera detector," https://tinyurl.com/jp3xl43, 2016.

[26] Workshop512, "Glint finder-camera detector," https://tinyurl.com/pbnfnur, 2013.

[27] FutureApps, "Hidden camera detector," https://tinyurl.com/y7nlhbx8, 2017.

[28] GalaxyApp, "Hidden camera detector pro," https://tinyurl.com/y745nqf7, 2016.

[29] Y. Cheng, X. Ji, X. Zhou, and W. Xu, "HomeSpy: inferring user presence via encrypted traffic of home surveillance camera," in *Proceedings of the 2017 IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 779–782, Shenzhen, 2017.

[30] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel traffic classification in the dark," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 229–240, 2005.

[31] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, p. 50, 2005.

[32] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli, "Revealing skype traffic: When randomness plays with you," in *Proceedings of the ACM SIGCOMM 2007: Conference on Computer Communications*, pp. 37–48, Japan, August 2007.

[33] M. Iliofotou, B. Gallagher, T. Eliassi-Rad, G. Xie, and M. Faloutsos, "Profiling-by-association: a resilient traffic profiling solution for the Internet backbone," in *Proceedings of the 6th International Conference on Emerging Networking Experiments and Technologies (Co-NEXT '10)*, Philadelphia, Pa, USA, December 2010.

[34] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang, "Internet traffic classification by aggregating correlated naive bayes predictions," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 5–15, 2013.