Reverse-Engineering the Physical Configuration of Smart Homes

Martin Vivian*, Ramin Sadre†
Institute for Information and Communication Technologies, Electronics and Applied Mathematics
UCLouvain
Belgium
Email: *martin.vivian@student.uclouvain.be, †ramin.sadre@uclouvain.be

***Abstract*—A major concern with cyber-security for Smart Homes are attacks against the privacy of their residents. Unfortunately, as has been shown in recent years, traditional cyber-security techniques, such a authentication, firewalls, and encrypted communication, are not sufficient to protect against privacy-invasion attacks. Researchers have shown that an attacker can detect the activities of the occupants by passively monitoring the encrypted wireless network traffic of the smart devices.**

In this paper, we focus on the question how the smart devices in a Smart Home are physically arranged. More specifically, we want to identify devices that are located in the same room or at least in close proximity. Similar to other works, we will only rely on information obtained through passive network measurements. We present techniques to determine whether a smart lamp and an IP camera are located in the same room. We then extend those techniques to cameras that have an automatic infrared mode switch. Finally, we show how the relative position and orientation of multiple cameras can be estimated.

***Index Terms*—IoT, Smart Home, activity detection, privacy, traffic monitoring**

I. Introduction

The Internet of Things (IoT) has turned into an immense success in the past years. Nowadays, IoT devices and applications can be found everywhere, in the streets (Smart Cities), in factories (Industrial IoT and Industry 4.0), in homes (Smart Homes), and soon also in interconnected self-driving cars. Not entirely unexpectedly, this success has also attracted the interest of malicious actors. We have witnessed more and more attacks against IoT systems in the past years. Among those, one of the most spectacular ones was the Mirai malware that infected probably millions of networked devices and turned them into members of a botnet, in this way misusing them to attack other Internet hosts and services [1].

With the raising popularity of IoT in Smart Homes, it can be expected that attacks against IoT systems will not only increase in frequency, but also become more harmful with an impact beyond the virtual world of the Internet. Similar to attacks against industrial control systems, an attack against a Smart Home can cause physical damage, for example by disturbing the control of a heating system. Another major concern of Smart Home security are attacks against the privacy of the residents. An attacker who gains control over devices in a Smart Home gets access to sensitive data that concerns the private life of the residents. An IP camera is the most drastic example of a privacy-sensitive device [2], but more humble devices can be transformed into spying instruments, too: a smart lamp with a motion sensor could tell when the house owners are at home, a smart thermometer when they go to bed; a fitness tracker reveals information about the health of the person wearing it, etc.

Unfortunately, as has been shown in recent years, traditional cyber-security techniques, such as authentication, firewalls, and encrypted communication, are *not* sufficient to secure Smart Homes against privacy-invasion attacks. Researchers have repeatedly demonstrated that attackers can obtain information about the activities of the members of a Smart Home household just by passively monitoring ("sniffing") the encrypted wireless network activities of the smart devices [3], [4], [5], [6], [7], [8]. To achieve this, researchers use machine learning to train activity detection systems on labeled traffic datasets. In that way, a detection system can for example learn that a specific traffic pattern stands for a triggered door sensor. This can be extended to more complex human-device interactions: entering a room results in network activities first of the door sensor and then of a motion sensor in the same room; a system trained on a suitable dataset can conclude that a person has entered the room if it observes the network activities of those two devices in the above order.

In this paper, we look at a related problem. We want to know how the smart devices are physically arranged in a Smart Home. More specifically, we want to identify devices that are located in the same room or at least in close proximity. Similar to the above works, our goal is to obtain this information from passive network measurements. The knowledge gained in this way can be used for benign purposes, such as troubleshooting or localizing forgotten or unregistered devices. But it could be also used my malicious parties. For example, they could learn whether there are rooms that are monitored by several cameras (e.g. a visible one and a hidden one). Combined with external observations of the Smart Home (e.g. illuminated windows), they could determine which rooms to avoid for a burglary or which rooms might contain valuable objects.

Our approach could be applied to various kind of devices. However, in this paper, we focus on smart lamps and IP cameras. Concretely, we first present an algorithm to detect whether an IP camera and a smart light are in the same room. We then introduce an alternative detection approach for

cameras with an automatic infrared mode. Finally, we show how the relative position and orientation of multiple cameras can be estimated.

The remainder of the paper is organized as follows. Related work is discussed in Section II. Our approach is presented in Section III. We present the results of our experiments and a discussion of the limitations of our approach in Section IV. The paper is concluded in Section V.

II. Related work

A. Device fingerprinting

In our adversary model (see Section III-A), we assume that the attacker is able to identify the type of device from passive network measurements. Many methods to do this can be found in the literature. Simple methods based on techniques such as MAC address analysis or analysis of DNS queries are described in [9], [10]. Methods applying machine learning algorithms on observations of the IP address, the protocols, the ports, or the packet lengths can be found in [11], [12], [3], [13]. More specific to IP cameras, Li *et al.* describe techniques to recognize typical fluctuations in the network traffic caused by video compression [7] . This is possible for some codecs, like for example H.264 that generates distinct patterns of I frames and P frames in the packet stream. Finally, Amar *et al.* [10] analyze data collected at a testbed consisting of various types of devices, including Apple TV, Hue Bridge, and Amazon Echo. They establish statistics on the typical communication of these devices, such as their consumption of bandwidth, the frequency of communication with other services like Dropbox services, etc. All this information can be used to derive a unique fingerprint of a device that helps to identify it later.

B. Activity detection

Here, activity detection refers to the problem of inferring device (or human) activities in a Smart Home from observations of the device's network traffic.

Acar *et al.* [3] determine the state of IoT devices (for example, if they are on or off) by using different machine learning classifiers like Random forest. With this information, they are able to recognize some activities that involve a state change. They also discuss time-depend activities. For example, someone who walks from room A to room B will trigger device activities that appear in a specific order. Zhan and Haddadi [14] identify the trajectory of multiple people in a house from measurements, provided the position of the sensors in the house is known. Their approach uses machine learning algorithms and graph theory.

Many IoT devices in Smart Homes are controlled by smartphone applications. Junges *et al.* [6] show that activities can be recognized by looking at the packets sent by the smartphone to an IoT Gateway. They do that by analyzing the number of bytes of the commands sent by the user. This approach has been tested on smart lamp holders and smart plugs.

Cheng *et al.* [4] describe an algorithm to detect human movement by monitoring the traffic rate of the IP camera. They implement their algorithm in an Android application running on a smartphone with a WiFi-card in monitor mode. The latter allows them to eavesdrop wireless network traffic without joining the network. Moreover, they are able to identify the camera device. Their *CUSUM* approach used by their algorithm to detect movements is also used by us (see Section III).

Li *et al.* [7] present a technique to identify the type of movement a person does in front of a camera. They recognize activities such as eating, dressing, moving, styling hair, etc. To this end, they observe the traffic size evolution over time. They divide the traffic timeseries into multiple segments by employing Binary Segmentation. For example, to recognize eating, they find the segment that corresponds to getting food and the segment corresponding to chewing the food. In that way, the pattern of an eating activity is the concatenation of those two segments.

The influence of the luminosity on the traffic sent by an IP camera is studied by Wampler *et al.* [15]. They show for different camera brands that the traffic decreases when the light in the room is turned off. The reason for this is that a camera is not able to record scene details if the light is too low. This increases the compressibility of the image and, therefore, reduces the amount of traffic sent by the camera. They also observe that luminosity impacts the quality of the detection of a movement.

C. Defenses

It is difficult to defend against the attacks described above since they do not require access to the payload of the network traffic. Countermeasures proposed so far by the research community are not very practical since they either require modifications in the network and application protocols used by the devices or the injection of significant amounts of fake traffic into the Smart Home network to hide the real traffic from a malicious observer. Moreover, many of those countermeasures are simply not effective against sophisticated analysis techniques [3], [16], [17].

III. Approach

A. Adversary Model

We assume that the attacker is able to perform traffic rate measurements on the network used by the IoT devices. If devices use different networks or network technologies (e.g. WiFi and Zigbee), the attacker has to monitor all those networks. The monitoring happens in a completely passive way, i.e., the attacker can neither inject/modify traffic nor physically interact with the devices.

We also assume that the attacker is able to identify the type of the devices, i.e., which devices are IP cameras and which are lights. This is not difficult to do since IoT device types have distinct network fingerprints. For further details, we refer to the existing literature on device fingerprinting (see Section II for some examples).

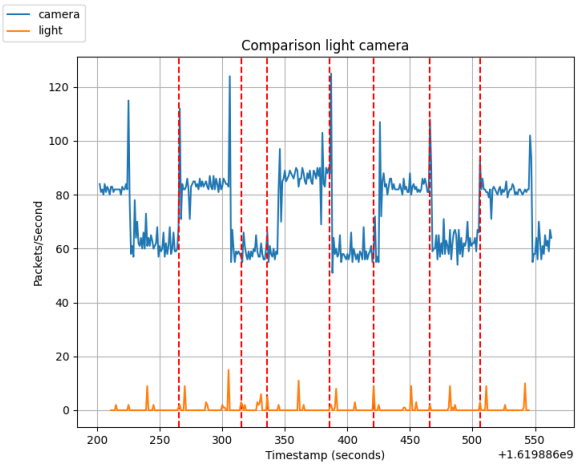

Fig. 1. Network traffic of camera and light device when light is repeatedly switched on or off

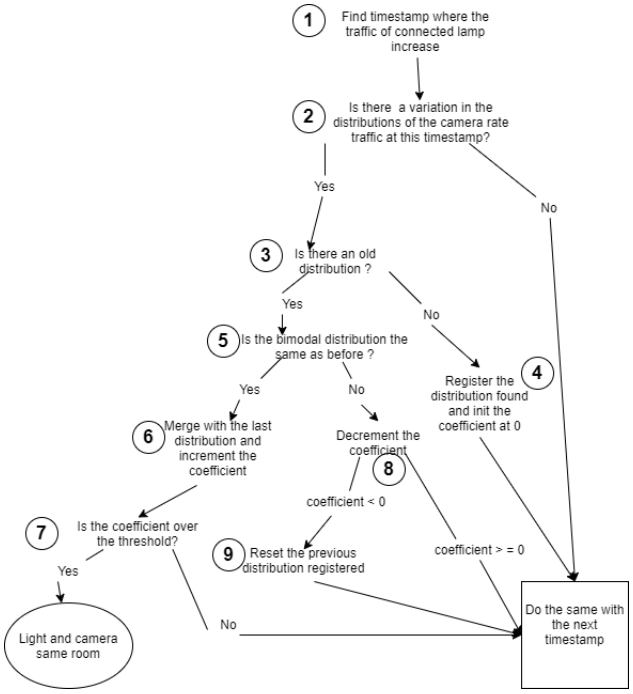

Fig. 2. Steps of algorithm to determine colocation of camera and smart light

B. Determining the colocation of devices

We start with a description of a new algorithm to determine whether an IP camera and a smart light are in the same room (or at least close enough that the light has an impact on the camera). Our approach is based on the observation made by Wampler *et al.* [15] (see Section II) that the brightness of a scene influences the amount of traffic sent by an IP camera per time unit. In fact, the camera's traffic rate (measured in the following in packets/s) follows a bimodal distribution with the larger (smaller) mode corresponding to the camera filming a lit (dark) room. Consequently, when the light in the room is turned on or off, this will cause a sudden change in the traffic rate of the camera in the same room, as shown in Figure 1. In addition, since we are using a smart light, whenever we turn it on or off, the light device itself will send or receive a certain number of network packets. Between those actions, no or nearly no traffic is transmitted to or from the light. However, as can be seen in Figure 1, the changes in the network traffic of the camera and the smart light are not perfectly synchronized due to delays in the network and the involved IoT services.

Our algorithm to determine whether the smart light and the camera are linked (in terms of cause and effect) is depicted in Figure 2. The algorithm takes as input the traffic rate timeseries of the camera and the light. The algorithm first finds all timestamps $t_1, t_2, \ldots$ with a network activity of the smart light (step 1). To reduce the number of occurrences, only one per 4-second time window will be counted. For each of those timestamps t_i, we compare the traffic of the camera in the window $[t_i - T, t_i]$ to the traffic in the window $[t_i, t_i + T]$ (step 2). The general idea is the following: If the average traffic rate in the two windows is different, it means that the camera is influenced by the light and we conclude that both devices are in the same room. However, as visible in Figure 1, the observed traffic rates are disturbed by noise caused by, among others, fluctuations in the network link quality. Therefore, instead of directly comparing the traffic rates, the algorithm performs a hypothesis test where the null hypothesis is that the traffic rates in the two windows are identical. To calculate the P-value, the algorithm uses Welch's t-test [18] ($\alpha = 0.05$). This test is an adaptation of the student test for the comparison of means of distributions with different variances. If the null hypothesis is rejected the algorithm assumes that the camera's behavior has changed.

In practice, basing our decision on the windows $[t_i - T, t_i]$ and $[t_i, t_i + T]$ of a single timestamp t_i would not be very reliable. The risk of false detection of a change in the camera's data traffic is too high. To make the detection more robust, our algorithm keeps track of the observed bimodal distributions of the camera's traffic at the timestamps $t_1, t_2, \ldots$ and only decides that the light and the camera are linked if those distributions are consistent over several timestamps. This happens in steps 3 to 9 in Figure 1. For the first timestamp, the observed bimodal distribution is just stored and a "consistency coefficient" variable is set to 0 (steps 3 and 4). For a subsequent timestamp t_j, we test whether the bimodal distribution observed in the windows $[t_j - T, t_j]$ and $[t_j, t_j + T]$ is identical to the one observed so far for the previous timestamps (step 5). Again, we use Welch's t-test. If they are identical, the distributions are merged and the coefficient variable is incremented by one (step 6). If it reaches a certain threshold θ (step 7), the algorithm concludes that the light and the camera are in the same room. However, if the bimodal distributions are not identical, the coefficient is decremented by one (step 8), and if the new observations are too different from the old ones, the history is completely deleted (step 9).

The value of the threshold θ controls how often the algo-

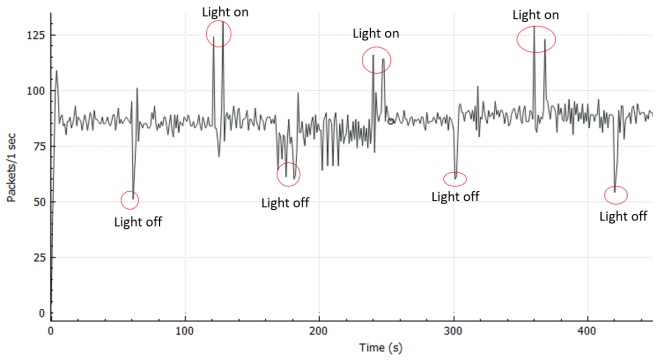

Fig. 3. D-Link camera traffic with automatic infrared night vision

rithm must see a change in the traffic rate of the camera before it decides that the camera's behavior is linked to the light. In Figure 1, the dashed vertical lines represent the timestamps when the algorithm reaches step 6, i.e. when the coefficient is incremented. Note that the first observed traffic rate change at timestamp 220 does not increment the coefficient (step 4).

C. Multiple cameras or lights

The algorithm described in the previous section can be extended to Smart Homes with multiple cameras or lights in the same room. To decide whether two cameras are in the same room, we check whether the detected traffic rate changes happen at (more or less) the same time.

On the other hand, if the algorithm does not find any light that could explain a camera's behavior, this either means that (a) there is no artificial light near the camera (e.g. because the camera is outside), (b) there is an artificial light source but it is not a smart light, or (c) there is a smart light but its network traffic cannot be observed. Case (a) and (b) can be distinguished by looking for the presence of the typical bimodal distribution in the camera's traffic. To confirm/exclude case (c), the attacker would have to extend the network monitoring to other network types (e.g. the camera uses WiFi, while the light connects to a wired Zigbee hub) to make sure that no traffic is missed.

D. Cameras with auto infrared light

Many IP cameras have built-in infrared illumination. Typically, the infrared light is automatically switched on when the room becomes too dark. We tested our algorithm from Section III-B with two cameras and discovered that the network traffic of one of them, a D-Link camera, does not show anymore a bimodal distribution when its automatic infrared mode is activated. Instead, it shows two distinct traffic patterns of around five seconds duration when the smart light in the room is switched on or off (see Figure 3).

To detect these patterns in the traffic we use a sequence search. A similar approach has been followed by Fu *et al.* to recognize activities of smartphone applications. First, we take the traffic rate timeseries of the camera containing the patterns of interest. Then, we compute a new timeseries with the rate change, i.e. the value at time t is the difference between the traffic rate at time t and at time $t-1$. We normalize and quantize the values in the new timeseries to a smaller set of labels, e.g. rate changes between 0 and 10 packets/s will get the label '1', rate changes between 11 and 20 packets/s get the label '2' and so on. Finally, we apply the Spade algorithms [19] for frequent sequence mining to the series of labels. Once the sequences representing the light-on/off patterns have been identified, we can replace step 2 in our algorithm by a comparison of the observed traffic with the identified sequences. As before, we deduce that the camera and the smart light are in the same room if the behavior of the camera matches the network activity of the light.

While this approach allows us to analyze the traffic of cameras that do not show the bimodal traffic distribution, it has the disadvantage that it requires a training dataset from which the sequences can be extracted. The sequences are camera specific, which means that not only do we have to prepare a separate dataset for each camera model, we also have to correctly identify the camera model from the Smart Home traffic in order to know which sequence set to use. Fortunately, the latter is not a big hurdle given the existing literature on how to do accurate device fingerprinting (see Section II).

E. Multiple cameras and human movement

The goal of the following technique is to determine whether and how two cameras film the same human movement. To this end, we use the CUSUM formula by Cheng et al [4]. It is part of a sequential analysis technique used for monitoring change detection. Since we are looking for an increase in traffic (movements increase the amount of traffic sent by a camera), we need to calculate the upper bound S_n of the CUSUM at time n:

$$S_0 = 0, \qquad S_n = max(0, S_{n-1} + x_n - w_n)$$

where w_n is the mean of the traffic rate, S_{n-1} is the CUSUM value at time $n-1$, and x_n is the current traffic rate at time n. In our code, we implement a modified version $S_n = max(0, S_{n-1} + x_n - w_n - \frac{k}{2})$ where k is the standard deviation of the traffic rate. This additional term allows the CUSUM value to decrease faster and stay at 0 when there is no significant increase in the traffic rate. With this technique we can rapidly detect if someone moves in front of a camera.

In Figure 4, we show the positive bound of the CUSUM calculated for two cameras that film the same person who walks back and forth in an apartment. The filled rectangles at the top of the figure represent the time periods when the CUSUM values of a camera exceeds a certain threshold (that we have set close to zero since we want to detect all movements). The fact that those time periods of movement of camera 1 do not significantly overlap with those of camera 2 indicates that the two cameras are not filming the same scene. In fact, both cameras were placed by us in different rooms for this experiment. Conversely, a high degree of overlap would indicate that the fields of view of the two cameras overlap, and the higher the overlap the more likely the two cameras

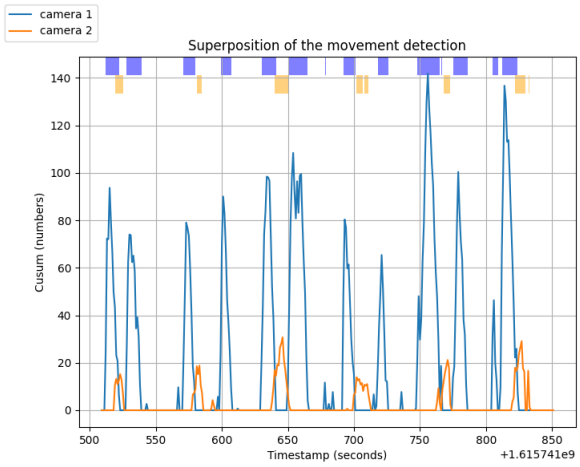

Fig. 4. Crossing room A, reaching room B, and coming back in the opposite direction. Repeated six times.

are near each other or even located in the same room. For the experiment in Figure 4, the overlap is only 7 percent.

We can estimate the "distance" between the two cameras by calculating the time lag between the two CUSUM timeseries. Note that this distance does not directly represent the physical distance between the positions of the cameras. Instead, it represents the amount of overlap of their fields of view. If it is zero, it indicates that the cameras are monitoring more or less the same part of the room, although not necessarily from the same direction. A lag close to the width of a peak in the CUSUM curve means that the cameras monitor adjacent rooms or adjacent parts of a room.

We find the lag by calculating the the cross-correlation of the two CUSUM timeseries. The lag corresponds to the time by which we have to shift one of the timeseries to maximize the cross-correlation.

IV. Experimental Results

In this section, we validate the techniques presented in Section III and present the experimental results.

A. Setup

The setup for the experiments is shown in Figure 5. Two cameras are connected to a router/WiFi-AP (router 2). Two smart lamps communicate over Zigbee with a Hue bridge wire-connected to the same router. Router 1 provides Internet connectivity, so that the IoT devices can reach their services deployed in the cloud. The cameras stream their videos to apps running on a smartphone connected to router 1. Between router 1 and router 2, we have placed a notebook that records the incoming and outgoing WiFi traffic. The following list contains the used hardware:

- **D-Link Mini HD Wi-Fi dcs-8000lhe:** WiFi camera with sound, infrared night vision;
- **Xiaomi Mi Home Security Camera 360° 1080p:** WiFi camera with sound sensors, infrared night vision, loudspeaker, motor for rotation;

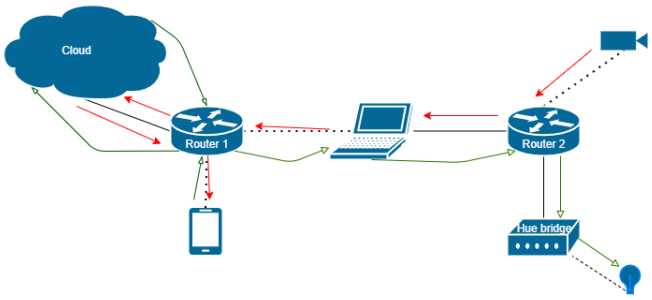

Fig. 5. Configuration to collect the packets data of the IoT devices

TABLE I
Scenarios

Scenario	Room A	Room B	Room C
1	lamp, cam_1	lamp, cam_2	None
2	lamp	lamp cam_1, cam_2	None
3	lamp, cam_1	lamp	cam_2
4	lamp, cam_2	lamp	cam_1

- **Hue bridge:** Zigbee bridge for the lamps;
- **Philips Hue bulb E27:** connected lamp whose brightness can be controlled by a smartphone application;
- **Lenovo Ideapad 320-15IKB** with Windows 10;
- **Technicolor TG799TSvn V2** router.

The traffic generated by the two cameras takes different paths depending on the camera model. The D-Link camera sends its video stream directly to the smartphone. In contrast, the video stream of the Xiaomi camera is first sent to the cloud and then returns to router 1 to finally reach the smartphone. When we control the lamp with the phone the traffic passes through router 1 and router 2. Note that all traffic is visible to the notebook placed between the two routers.

B. Lamps and cameras without infrared mode

To evaluate the performance of the algorithm from Section III-B, we test four scenarios without natural light where we place the devices in three different rooms of a house (see Table I). The infrared night vision is not activated. Scenario 3 and 4 allow us to test the robustness of the detection in terms of false positives since room B only contains a lamp and room C only contains a camera. Figure 6 shows how we control the lamps during the experiments. For each scenario, we repeat the depicted control pattern for 6 minutes. We test all four scenarios in two different houses called 'Location 1' and 'Location 2' in the following.

We choose a window size of $T = 15s$. This is a compromise between detection latency and detection quality. A larger T

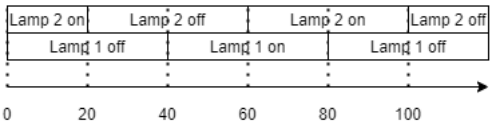

Fig. 6. Time schedule (in seconds) for lamps turned on/off

TABLE II
RESULTS AFTER STEP 1 AND 2

	Location 1		Location 2	
	Xiaomi	D-Link	Xiaomi	D-Link
Accuracy	0.56	0.71	0.5	0.73
Precision	0.78	0.77	0.88	0.86
Recall	0.67	0.91	0.54	0.83
F1-Score	0.72	0.83	0.67	0.84

means to wait longer before the algorithm can be executed. On the other hand, more data is available for the distribution estimation. Another limiting factor is the expected time between two light changes. In our experiment, we switch a light on or off after 40 seconds. Obviously, T should be shorter than that.

When we use a relatively conservative threshold of $\theta = 4$, the algorithm correctly identifies without any errors, for all scenarios and locations, whether a specific lamp and camera are located in the same room.

To get a better insight into the performance of the algorithm, we have also analyzed the results of just step 1 and step 2 of the algorithm, i.e., what would happen if we decided whether a lamp and a camera are in the same room just based on the tests in step 1 and 2, without performing the additional tests in steps 3–9. Table II shows for each location and camera the results obtained from the four scenarios. We observe that especially the accuracy for the Xiaomi camera is low. In general, the traffic of this camera has a higher variability than the other camera, which makes the detection more difficult.

C. Auto infrared with sequence search method

We have repeated the same four scenarios with the automatic switch to infrared night vision activated. As already explain in Section III-D, the Xiaomi camera keeps its binomial distribution and the results reported in the previous section do not change. For the D-Link camera, we first create a pcap trace of 9 minutes where we have a lamp and the camera in the same room in Location 1 and we turn on/off the light every 30 seconds. We run the sequence mining algorithm on the trace to identify the patterns that characterize the changes of the state of the camera. Table III shows the results. We observe that they are similar to the results obtained without infrared night vision. Like in the previous experiments, the results demonstrate that a single test is not sufficient; multiple tests and a consistency threshold are needed to obtain a reliable detection.

Figure 7 shows an example of the pattern recognition. The dashed vertical lines indicate the timestamps when the switch of the infrared mode has been detected.

D. Human movements

To illustrate the general behavior of the cameras in the presence of movements, we show in Figure 8 the resulting traffic rate in a situation where someone walks in front of the two cameras for 20 seconds and repeats that every minute. As can be seen, both cameras react to the movement, but

TABLE III
RESULTS AUTOMATIC SWITCH TO INFRARED NIGHT VISION ON THE D-LINK CAMERA

	Location 1	Location 2
Accuracy	0.74	0.69
Precision	0.95	0.86
Recall	0.77	0.78
F1-Score	0.85	0.82

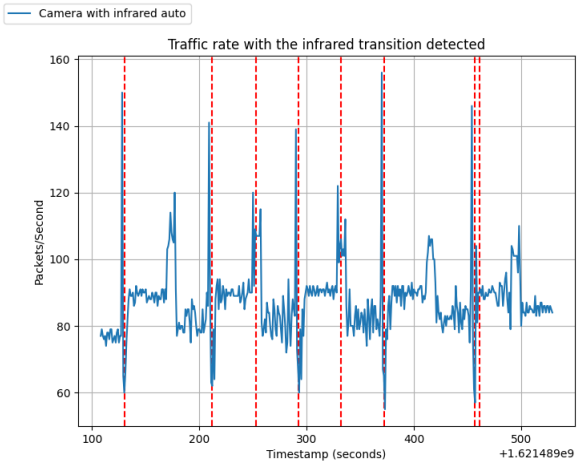

Fig. 7. Infrared switch recognition

with different traffic rates. Figure 9 shows the corresponding CUSUM curves. The filled rectangles at the top of the figure visualize the movement detection by our algorithm. The more the rectangles of the two cameras overlap, the more likely it is that the two cameras are seeing the same scene.

Figure 10 and Figure 11 show the traffic rate and CUSUM curves obtained in a scenario where someone repeatedly crosses the field of view of the cameras at walking speed.

Table IV shows the amount of overlap of the movement detection between the two cameras for different scenarios. In

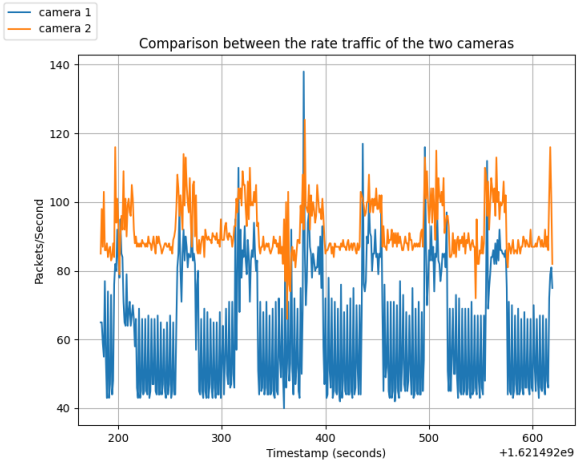

Fig. 8. Traffic rate: Walking for 20 seconds in front of the camera

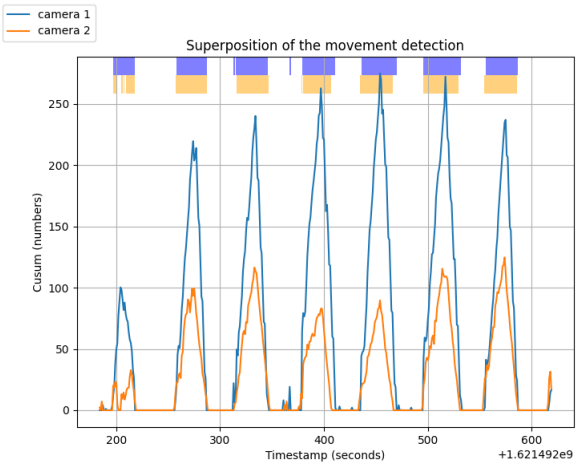

Fig. 9. CUSUM: Walking for 20 seconds in front of the camera

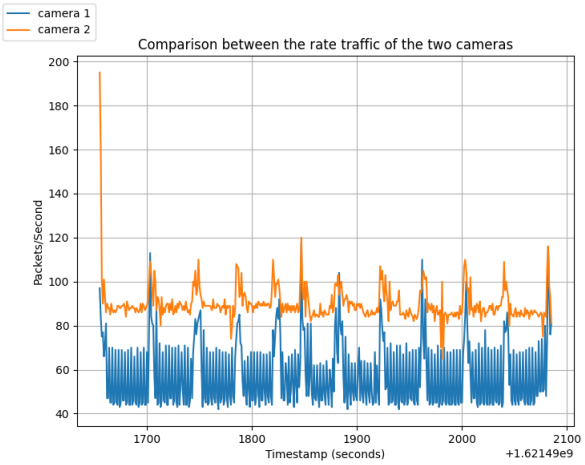

Fig. 10. Traffic rate: Crossing a room

the first scenario, both cameras are in the same room with the same field of view, and a person walks in front of them during 20 seconds. In the second scenario, both cameras are again in the same room and a person crosses their field of view during two to three seconds. In the last scenario, the cameras are in two adjacent rooms that are both crossed by the person. Each scenario was tested for 7 minutes with natural light. As expected, the shorter the movement, the smaller the overlap. If the cameras are in different rooms, the overlap is even smaller.

TABLE IV
PERCENTAGE OF OVERLAP BETWEEN THE TWO CAMERAS

Scenario	Location 1	Location 2
Walking for 20s in front of the cameras	85%	94%
Crossing a room in front of the cameras	57%	82%
Crossing two rooms in front of the cameras	25%	35%

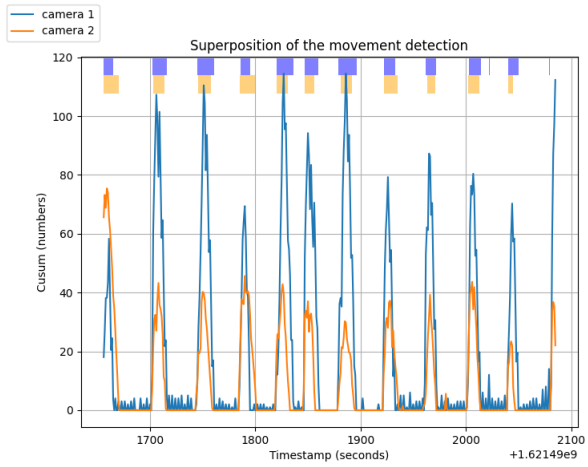

Fig. 11. CUSUM: Crossing a room

E. Orientation of the cameras

In this last experiment, we let a person cross a room monitored by two cameras. Forty seconds later the person crosses the room in the opposite direction. The cameras have been oriented so that each one films a part of the room and that the two cameras together cover the entire room. The movement detection results are shown in Figure 12. We observe that one time out of two, one of the two camera detects the movement first. But we still get a large overlap of the movement detection regions of the two cameras. That was not the case when we had two rooms with one camera in each one. We can also numerically estimate the lag between the two cameras as described in Section III-E. We calculate a mean positive lag of 2.8 seconds for the time periods when the person crosses the room in one direction and a mean negative lag of 3.2 when the person crosses the room in the other direction. This lag corresponds to the time it approximately takes to cross the part of the room only seen by one camera before being sighted by the other camera. Note that this method could also be applied to estimate the distance between two rooms if the cameras were located in different rooms.

F. Discussion and limitations

1) The stability of the network: As already mentioned above, the stability of the network has a significant impact on the quality of the results. A bad connection will make it more difficult to recognize some patterns and the speed at which we could recognize them. For example, a sudden drop in the network bandwidth could cause a change in the distribution of the camera traffic that could be misinterpreted. The algorithm the most sensitive to the network quality is the infrared sequence recognition. In a stable network, we are even able to reliably distinguish the switching off of a lamp from when it is switched on. But since we cannot assume that we always have good network conditions, we have limited the presentation and discussion of the results in Section IV-C to

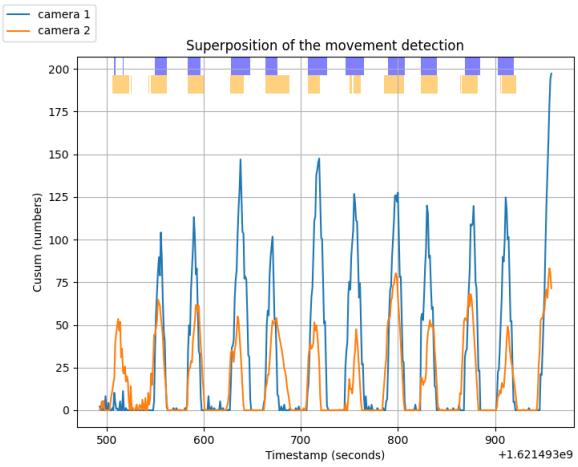

Fig. 12. Crossing a room monitored by two cameras oriented in different directions

the question whether a state change (on to off or vice versa) can be reliably detected.

2) The sensitivity of the camera to the light: In our four scenarios with lights and lamps, we eliminated all other light sources, i.e. the rooms were almost completely black when the smart lamp was switched off. We have also tested less dark rooms and discovered that the different sensitivity of the cameras to low light conditions can become a challenge that leads to wrong results. Concretely, if one of the cameras has a higher sensitivity than the other, it will show greater traffic variations when light conditions change than the less sensitive camera. Our algorithm might not understand this situation correctly and conclude that the cameras are in different rooms. We have observed such problems with the Xiaomi camera and, indeed, it has a higher light sensitivity than the D-Link camera.

3) Number of people: Our analysis of traffic variations caused by human movements assumes that only one person is present. If the cameras film multiple people, wrong conclusions on the locations of the cameras are likely. Similarly, this will affect the estimation of the proximity of cameras using the cross-correlation method.

4) Limited set of hardware: Both our cameras use the UDP protocol. In addition, we have only tested one smart light brand. As future work, we have to validate our results with other protocols and devices.

V. Conclusion

Privacy is a major concern of Smart Home security. In this paper, we have presented techniques that allow to infer the physical arrangement of smart devices in a Smart Home from passive network measurements. More specifically, we have presented algorithms to detect whether smart lamps and smart cameras are located in the same room. We have also shown how the relative position and orientation of multiple cameras can be estimated.

We have evaluated our techniques in different scenarios and shown that they provide results that give an insight into the physical organization of a Smart Home. We argue that such techniques can be useful for troubleshooting and management purposes, but could be also misused by malicious parties.

References

[1] A. Shoemaker, "How to identify a mirai-style ddos attack," https://www.imperva.com/blog/how-to-identify-a-mirai-style-ddos-attack/, accessed: 2021-05-27.

[2] Y. Seralathan, T. T. Oh, S. Jadhav, J. Myers, J. P. Jeong, Y. H. Kim, and J. N. Kim, "Iot security vulnerability: A case study of a web camera," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2018, pp. 172–177.

[3] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: I see your smart home activities, even encrypted!" in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020, pp. 207–218.

[4] Y. Cheng, X. Ji, X. Zhou, and W. Xu, "Homespy: Inferring user presence via encrypted traffic of home surveillance camera." in *ICPADS*, 2017, pp. 779–782.

[5] B. Copos, K. Levitt, M. Bishop, and J. Rowe, "Is anybody home? inferring activity from smart home network traffic," in *2016 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2016, pp. 245–251.

[6] P.-M. Junges, J. François, and O. Festor, "Passive inference of user actions through iot gateway encrypted traffic analysis," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 7–12.

[7] H. Li, Y. He, L. Sun, X. Cheng, and J. Yu, "Side-channel information leakage of encrypted video stream in video surveillance systems," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[8] W. Zhang, Y. Meng, Y. Liu, X. Zhang, Y. Zhang, and H. Zhu, "Homonit: Monitoring smart home apps from encrypted traffic," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1074–1088.

[9] N. Apthorpe, D. Reisman, S. Sundaresan, A. Narayanan, and N. Feamster, "Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic," *arXiv preprint arXiv:1708.05044*, 2017.

[10] Y. Amar, H. Haddadi, R. Mortier, A. Brown, J. Colley, and A. Crabtree, "An analysis of home iot network traffic and behaviour," *arXiv preprint arXiv:1803.05368*, 2018.

[11] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2177–2184.

[12] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Iotsense: Behavioral fingerprinting of iot devices," *arXiv preprint arXiv:1804.03852*, 2018.

[13] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: a machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the symposium on applied computing*, 2017, pp. 506–509.

[14] Y. Zhan and H. Haddadi, "Mosen: Activity modelling in multiple-occupancy smart homes," *arXiv preprint arXiv:2101.00235*, 2021.

[15] C. Wampler, S. Uluagac, and R. Beyah, "Information leakage in encrypted ip video traffic," in *2015 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2015, pp. 1–7.

[16] N. Apthorpe, D. Y. Huang, D. Reisman, A. Narayanan, and N. Feamster, "Keeping the smart home private with smart (er) iot traffic shaping," *arXiv preprint arXiv:1812.00955*, 2018.

[17] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, "Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail," in *2012 IEEE symposium on security and privacy*. IEEE, 2012, pp. 332–346.

[18] B. L. Welch, "The generalization ofstudent's' problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1/2, pp. 28–35, 1947.

[19] M. J. Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Machine learning*, vol. 42, no. 1, pp. 31–60, 2001.