Title
I Always Feel Like Somebody's Sensing Me! A Framework to Detect, Identify, and Localize Clandestine Wireless Sensors

Author
Singh, Akash Deep

Publication Date
2020

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

I Always Feel Like Somebody's Sensing Me!
A Framework to Detect, Identify, and Localize Clandestine Wireless Sensors

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Electrical and Computer Engineering

by

Akash Deep Singh

2020

ABSTRACT OF THE THESIS

I Always Feel Like Somebody's Sensing Me!
A Framework to Detect, Identify, and Localize Clandestine Wireless Sensors

by

Akash Deep Singh
Master of Science in Electrical and Computer Engineering
University of California, Los Angeles, 2020
Professor Mani B. Srivastava, Chair

The increasing ubiquity of low-cost wireless sensors in smart homes and buildings has enabled users to easily deploy systems to remotely monitor and control their environments. However, this raises privacy concerns for third-party occupants, such as a hotel room guest who may be unaware of deployed clandestine sensors. Previous methods focused on specific modalities such as detecting cameras, but do not provide a generalizable and comprehensive method to capture arbitrary sensors which may be "spying" on a user. In this work, we seek to determine whether one can walk in a room and detect *any* wireless sensor monitoring an individual. As such, we propose SnoopDog , a framework to not only detect wireless sensors that are actively monitoring a user, but also classify and localize each device. SnoopDog works by establishing causality between patterns in observable wireless traffic and a trusted sensor in the same space, e.g., an inertial measurement unit (IMU) that captures a user's movement. Once causality is established, SnoopDog performs packet inspection to inform the user about the monitoring device. Finally, SnoopDog localizes the clandestine device in a 2D plane using a novel trial-based localization technique. We evaluated SnoopDog across several devices and various modalities, and were able to detect causality 96.6% percent of the time, classify suspicious devices with 100% accuracy, and localize devices to a sufficiently reduced sub-space.

The thesis of Akash Deep Singh is approved.

Danijela Cabric

Xiang Chen

Mani B. Srivastava, Committee Chair

University of California, Los Angeles

2020

Table of Contents

List of Figures . . . . . **vii**

Acknowledgments . . . . . **ix**

Curriculum Vitae . . . . . **x**

1 Introduction **1**

2 Background and System Model **5**

2.1 System Model 6

2.2 Adversary Model 7

3 SnoopDog Overview **9**

4 Detecting and Identifying Snooping Wireless Sensors **11**

4.1 Searching for Wireless Sensors 11

4.2 Detecting Causality with User Activity 12

4.3 Characterizing a Representative Set of Snooping Sensors 13

4.4 Device Identification via MAC Address 15

5 Snooping Sensor Localization **17**

5.1 Identifying Sensor Coverage 17

5.2 Ensuring Sufficiently Reduced Region 19

6 Implementation **21**

6.1 Experimental Setup 21

6.2 Overview 22

6.3 Aggregation of Traffic Statistics . . . 22
6.4 Detecting the Cause-Effect Relationship between User Motion and Hidden Devices . . . 23
6.4.1 Wireless Sensors that Encode Raw Data . . . 23
6.4.2 Wireless Sensors that Encode Inferred Events . . . 25
6.4.3 Device ID via MAC Address Lookup . . . 27
6.5 Device Localization . . . 29

7 Evaluation . . . 31
7.1 Wireless sensors that encode raw data . . . 31
7.2 Wireless sensors encoding inferred events . . . 32
7.3 Localization . . . 33
7.4 Overhead Analysis . . . 34
7.5 Effects of range on detection . . . 35
7.6 False negatives for background detection . . . 35

8 Discussion . . . 37

9 Related Work . . . 40
9.1 Conclusion . . . 42

Bibliography . . . 43

A Audio-based Localization for Personal Home Assistants . . . 47

B Techniques to fool SnoopDog . . . 48
B.1 No Encoding or Data Padding . . . 48
B.2 Adding Random Noise to the Data . . . 49

B.3 Constantly Vary the Resolution of the Data Being Transmitted 49

B.4 Adding a tape/broadcast delay to the transmissions 50

List of Figures

2.1 Overview of SnoopDog framework. The SnoopDog framework first identifies if a user is being monitored based on the cause-effect relationship between the values of a trusted sensor, e.g., an IMU, and traffic patterns. It then inspects the associated packets and identifies the possible devices based on the physical (MAC) address. Finally, SnoopDog localizes each device relative to the user based on the received signal strength indicator (RSSI) values. 8

4.1 I-P-B Frames [41] 14

6.1 Detecting frame rate of the camera. In this case, the frame rate of the camera is 25 Hz which is where the peak is. 23

6.2 traffic captured from a camera over a static scene and a scene where a human is walking around. 25

6.3 traffic of a camera and its comparison with IMU data of the user who was being monitored in the scene. 26

6.4 Modeled traffic for an RF sensor in a static scene and one where a user performs our detection trial. 27

6.5 28

6.6 28

6.7 (a) traffic of a motion sensor. The red-dotted line signifies a motion event. (b) traffic of an Alexa device for the user repeating the same phrase 4 times. 28

7.1 Lab dimensions and results of the detection trials. 33

7.2 A walk-through of the trial-based localization algorithm in the laboratory environment in Figure 7.1. The arrows represent the direction the laptop screen was facing. 34

7.3 traffic of a camera and its comparison with IMU data of a user walking randomly with stops 36

A.1 Trial-based localization for acoustic sensors. 47

B.1 Padding the motion sensor and the camera traffic 49

B.2 Injecting noise in the traffic of a motion sensor to fool SnoopDog 50

Acknowledgments

I would like to express my sincere gratitude to my advisor Prof. Mani Srivastava who was patient and motivational during the course of my M.S. study while also providing support and guidance wherever necessary. I could not have imagined having a better mentor for my M.S. study.

I would also like to thank the rest of my thesis committee: Prof. Danijela Cabric and Prof. Xiang 'Anthony' Chen for their insightful feedback and helpful discussions.

My sincere thanks also go to my labmates Dr. Luis Antonio Garcia and Joseph Noor, who collaborated with me and brainstormed a lot of ideas during the course of my thesis. Their insights were great in improving the quality of my thesis. I would also like to thank my fellow labmates at Networked & Embedded Systems Laboratory.

I will forever be grateful for the unwavering support and continuous encouragement provided to me by my parents and my sister. This thesis and all my accomplishments would not have been possible without them.

Finally, I would like to acknowledge the funding agencies that sponsored this work in part – the National Science Foundation (NSF) and the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

Curriculum Vitae

2014 – 2018	B.Tech (Honors) in Electronics and Communication Engineering, Indraprastha Institute of Information Technology, Delhi
2018 – Present	MS student in Electrical and Computer Engineering, University of California, Los Angeles (UCLA).

CHAPTER 1

Introduction

The explosion of internet-of-things devices in smart homes, buildings, and cities [37] can be partly attributed to the proliferation of low-cost wireless sensors in tandem with advancements in embedded device battery technology [12]. Affordable sensors, including cameras and motion sensors, have facilitated deployments to monitor and control these environments. Although there are profound positive impacts that ubiquitous sensor-rich environments can have on society, there is an inherent risk in enabling users access to such pervasive sensing – particularly when these environments host occupants oblivious to the presence of these sensors.

A person's physical privacy in these contexts is entirely at the discretion of the owner who deploys these sensors. Regulation is unclear in more informal settings, such as a guest residing in a home or a homestay lodging. Although these environments may be enhanced with a legitimate set of sensors and actuators to provide security, surveillance, comfort, and convenience, there have been several instances where a hosting owner has attempted to spy on the occupants in homestay settings [7], motel lodgings [15], and rooms aboard cruise ships [36]. There are even instances in well-established hotel chains and mall restrooms when a malicious employee or customer has bugged several rooms [31]. In [35], Southworthreport that such sensors are also used for 'intimate partner stalking', which may enable domestic abusers.

The prevalent method to detect bugs involves an RF receiver that senses if the received power in a particular frequency range is above a certain threshold. However, since bug detectors work on the principle of sensing surrounding RF signals, they can easily be falsely

triggered by legitimate RF devices such as mobile phones, radios, and other devices such as smart TV and smart doorbell in the vicinity. This lack of reliability limits the practicality of these detectors. Furthermore, they provide no semantic information regarding device information, location, or whether the device is actually monitoring a user. An alternate method has emerged to detect the presence of IoT devices based on network traffic statistics [14]; however, such an approach still fails to capture information about device location or active monitoring region.

More sophisticated solutions have recently been proposed to specifically detect wireless cameras. The general approach is to correlate known semantic information about the environment with network traffic patterns. For instance, Wampler [43] showed that changing lighting conditions causes notable variations to appear in a wireless camera's video traffic; that is, video encoding leaks sensitive environmental information. This discourse was leveraged to detect a camera by flickering a light source for a short period of time and correlating it to changes in network traffic [23, 28]. Similarly, an approach has been presented that correlates the traffic patterns of a trusted camera with traffic patterns of other hidden cameras on a network to detect whether they are simultaneously observing the same space [47]. But each of these camera-specific approaches, which correlate simultaneous observations between trusted cameras and hidden cameras, fail to generalize across modalities. For example, varying lighting conditions would be ineffective for detecting a hidden microphone or an RF sensor. More interestingly, there has been a preliminary effort that used human motion as to detect and coarsely localize hidden cameras [4]. Human motion is an example of an event that can be generalized across many modalities if the event is formalized correctly. Furthermore, human activity serves as the ideal reference event for determining whether a clandestine sensor is monitoring a human.

In this paper, we propose , a generalized framework to detect clandestine wireless sensors that are monitoring a user in a private space. SnoopDog leverages the notion of causality to determine if the values of a trusted sensor cause patterns in traffic stemming from other devices. In particular, SnoopDog works by having the user perturb the trusted sensor val-

ues to observe if there is a causal pattern in the traffic for a different device. For instance, if a wireless camera is monitoring a user who is wearing an inertial measurement unit (IMU), the IMU values indicate a causal relationship with the camera's traffic. SnoopDog utilizes encoding scheme models of different wireless sensing modalities to classify the sensor type, and then cross-references packet inspection with publicly available information of manufacturers to identify the specific device model. We further introduce a novel approach that leverages sensor coverage techniques to provide fine-grained localization of a detected sensor. We implemented SnoopDog utilizing a trusted set of sensors on a user's mobile phone as well as a packet sniffer to observe traffic patterns. In the future, we envision SnoopDog to be implemented as an app on either a smartwatch or a smartphone, both of which have sufficient sensing capabilities (with improvements in their cards that would allow them to hop channels in monitor mode) to make it easily accessible to non-technical users.

SnoopDog operates in three phases. Assuming the trusted set of sensors is on the user (e.g., a wearable device or smartphone), SnoopDog is first in a passive monitoring (background) phase, searching for suspicious causal patterns between the wireless traffic and the user's normal activity. If a device is flagged as potentially monitoring the user, an active phase is engaged, and the user is instructed to perform a series of specific actions to detect the sensor with high fidelity. Finally, if the sensor is unable to be spotted through a preliminary search, a localization phase engages for accurate ascertainment of clandestine placement. The user can either skip the background or the active phase per their convenience.

We evaluate SnoopDog over a representative set of wireless sensors following a taxonomy of popular sensing devices that may be used for surveillance. The framework had a detection rate of 96.6% and a device classification rate of 100% when the injected multi-modal event was human motion. We show that the location of the bug can be narrowed down to a sufficiently reduced region that facilitates the user's search for the device. This feature is a vast improvement over state-of-the-art approaches that localize devices as either indoors or outdoors. While SnoopDog cannot detect *any* wireless sensor monitoring the user (chapter 8), it can detect a broad set of commonly used wireless sensors. We further formalize the

challenges and limitations across different modalities.

Contributions: Our contributions are summarized as follows:

- We propose SnoopDog , a generalized framework to detect arbitrary hidden clandestine sensors by leveraging the cause-effect relationship between a trusted set of sensor values observing an injected event and traffic patterns.
- We present a novel technique that leverages the notion of directional sensor coverage to provide state-of-the-art localization for clandestine devices.
- We show how SnoopDog can be extended to identify the model of a device based on packet inspection and publicly available information of device manufacturers.
- We evaluate SnoopDog with a mobile phone and a packet sniffer on a representative set of clandestine sensors and show a detection rate of 96.6% and a device classification rate of 100% when the injected multi-modal event is human motion.

CHAPTER 2

Background and System Model

The general approach to detecting wireless sensors relies on the notion that the device's wireless communication leaks information in some domain. This aspect has been exploited for the development of wireless bug[1] detectors which can sense the presence of wireless transmitters in a space [32, 42]. Bug detectors are RF receivers that look for signals in a frequency range with a received power above a certain threshold. The received power threshold and frequency range can be set according to a target set of wireless devices. For instance, to detect sensors that communicate over , a device would scan frequency ranges around 2.4 GHz or 5 GHz. Similarly, the range can be set accordingly for other wireless technologies like Bluetooth [27] and Z-wave [6]. In tuning the received power threshold, there is a direct trade-off between detection accuracy and false positives [32]. If the threshold is too low, one may falsely attribute wireless signals from other devices in the space, like mobile phones, to bugs. On the other hand, a high threshold risks ignoring wireless bugs that are not within close proximity of the detector. As these detectors provide no semantic information about the detected signals, it is difficult to assume whether or not the observed signal is truly originating from a hidden bug [42].

As wireless sensors transmit their information via packets, another technique to detect them uses packet sniffing. Approaches like DewiCam [4] sniff wireless packets and use their characteristics to train a classifier to identify whether or not a particular device is a camera. However, even if the type of device is determined, it may or may not be monitoring the user. If there is a camera monitoring the door of a house, it does not pose the same threat to a

[1] A *bug* in this context refers to a hidden device spying on the user.

user's privacy as a camera that is monitoring the bedroom. Hence, even if we are able to detect what type of device is present in the space, it is difficult to characterize if its intention is adversarial. A direct way to identify whether a device poses a potential privacy threat is to determine whether or not it is actively monitoring the user.

Detecting sensors monitoring a physical space. If a wireless sensor is monitoring someone in a physical space, the data that it captures is a function of the person's interaction with the space. For example, if someone moves into a space monitored by a motion detector, the sensor's control mechanism may be triggered and begin uploading relevant information to the cloud to be processed and forwarded (e.g., an alert to the device owner or downstream actuation). Similarly, the information recorded by a video camera captures variation as a result of motion within the scene that it is capturing. If another sensor can observe and measure the interaction of the user with their surroundings, we can identify whether the user's actions indicate a causal relationship with the wireless traffic of the sensor. If such a relationship is found, then the sensor must be monitoring the user. To generalize our approach, we provide a system and an adversary model.

2.1 System Model

We consider a system model for SnoopDog where a user has access to a laptop or smartphone device with a network card that can enter monitor mode to sniff wireless packets over the same channel as one or more clandestine sensors. The system should further be equipped with a trusted set of *ground truth* sensors to establish causality between the sensor values and the associated patterns from the clandestine wireless sensor(s)[2]. These capabilities require a set of certain assumptions.

sniffing assumptions. We assume that the sniffer on the user's device can monitor the encrypted traffic streaming from the clandestine device. SnoopDog does not require any

[2]We assume there may be additional, non-clandestine sensors that are monitoring the user. Such superfluous information is still informative, as the goal of this work is to detect all wireless sensors monitoring a user.

form of granted access to a particular network, i.e., SnoopDog should be able to sniff the device regardless of whether or not the network is closed or hidden. Unlike previous solutions, this implies that the user does not need to know the SSID or password of the network.

Causality assumptions. We assume that the user has a sufficient set of trusted ground truth sensors whose modalities are sensing any of the user's activities that would exhibit a causality with the encoding patterns of any clandestine wireless sensors. We formalize the notion of sufficient causality in chapter 4.

2.2 Adversary Model

The adversary's goal is to remotely spy on a third-party occupant of a private space in real-time. We assume the adversary uses an arbitrary set of wireless, commercial-off-the-shelf (COTS) sensors that are tailored for clandestine placement. The communication between the attacker and sensor may be encrypted and placed on an arbitrary wireless frequency band. We further assume the adversary has deployed these clandestine sensors in a manner that is not apparently visible to the user within the space. We focus on an attacker utilizing devices that communicate over , as this is the most prevalent method of wireless communication for remote monitoring using commercial and consumer equipment[3].

[3]Although SnoopDog focuses on -connected devices, we discuss in chapter 8 how such a system could be generalized to other wireless communication standards and protocols.

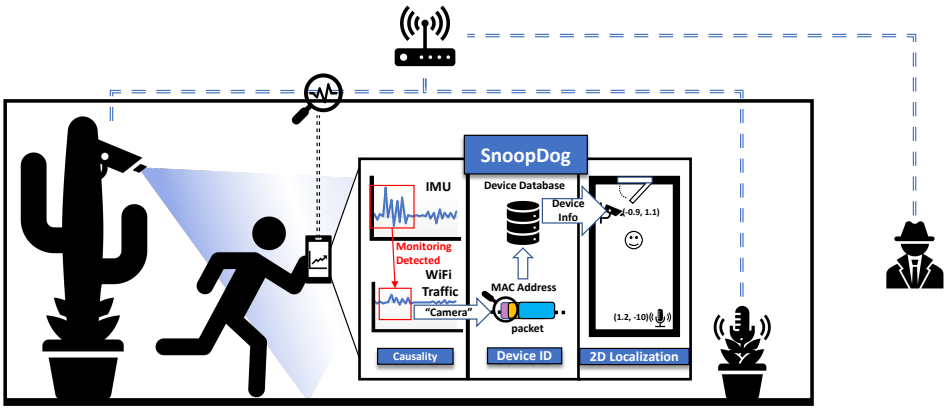


Figure 2.1: Overview of SnoopDog framework. The SnoopDog framework first identifies if a user is being monitored based on the cause-effect relationship between the values of a trusted sensor, e.g., an IMU, and traffic patterns. It then inspects the associated packets and identifies the possible devices based on the physical (MAC) address. Finally, SnoopDog localizes each device relative to the user based on the received signal strength indicator (RSSI) values.

CHAPTER 3

SnoopDog Overview

The goal of SnoopDog is to identify and localize clandestine wireless sensors within an arbitrary space. As depicted in Figure 2.1, SnoopDog can detect and localize a wireless sensor given it has access to a trusted sensor that can measure and quantify the ground truth in the modality that we are trying to detect. SnoopDog works in three phases. When a user first enters a new space, SnoopDog operates in a background mode to determine whether a user is being monitored based on the cause-effect relationship between the values of a trusted sensor (e.g., an on-body IMU) and traffic patterns. If the user wants to clear a room immediately, the background phase may be optionally skipped; alternatively, the background phase offers a low-overhead solution to bug detection. If a clandestine sensor is discovered, SnoopDog enters its second phase and asks the user to perform a unique perturbation in the space to further ascertain the presence of a snooping sensor. The associated packets are then inspected to identify the possible device type based on the physical (MAC) address. Finally, in the third phase, SnoopDog utilizes a trial-based localization technique to identify the specific placement of the monitoring device. With the appropriate selection of ground truth sensor, that is, a device which can semantically capture at least a subset of the events captured by the snooping device, SnoopDog can detect clandestine wireless sensors of arbitrary modality.

The objectives for a solution which can detect hidden devices in space should have the following characteristics:

- The solution must work for arbitrary sensing modality.
- The user must be able to generate events in the space that will establish causality

between a sufficient set of ground truth sensors and any clandestine sensors.

- The solution must work equally well in indoor and outdoor conditions.
- The solution must be reasonably compact enough for a user to easily transport from room to room.
- The solution must work for all configurations.
- The solution should not be affected by encryption.

Given these challenges, we present our design for clandestine wireless sensor detection, identification, and localization.

CHAPTER 4

Detecting and Identifying Snooping Wireless Sensors

This chapter outlines the ability of SnoopDog to detect whether a clandestine sensor is actively snooping on a user. We describe the search space for wireless sensors, how to establish causality, how to generalize across modalities, and how to understand various sensors' wireless transmission.

4.1 Searching for Wireless Sensors

The adversary can create a network and connect the snooping device to it. As a result, the hidden device can be present in any of the possible channels. Even though SnoopDog does not need access to these networks, it still needs to scan all frequencies and look for any devices transmitting on them. 2.4 GHz and 5 GHz are the most popular bands for networks, and as such, we focus on those particular bands, even though the SnoopDog scan region can be easily extended to include other ranges. During discovery, the Network Interface Card (NIC) scans through all channels sequentially to find available access points (APs) [13, 46]. Similarly, SnoopDog also scans through all the channels in monitor mode, but instead of looking for available APs, it looks for transmissions in those channels and creates a list of devices using the MAC address present in packet headers. As a result, SnoopDog does not need to be connected to any specific AP to operate. Even if a network is hidden, its transmissions can still be observed by monitoring the channel. Thus SnoopDog can detect devices on any network. Because devices may transmit data intermittently, SnoopDog continuously scans all channels and actively maintains an aggregate set of traffic data. Once the list of devices has been populated, SnoopDog then seeks to detect causality between

user activity and data being transmitted from each device.

4.2 Detecting Causality with User Activity

Detecting the cause-effect relationship between the action of a user in a space and the data captured by a clandestine, wireless sensor requires access to two essential components: 1) a ground truth sensor to capture information about the user in the space and 2) a representation of the data collected by the clandestine sensor. While data packets transmitted by wireless sensors may be encrypted, the header information is not. This header information provides us with the MAC address and payload size of each transmitted packet. This data can be grouped and aggregated for all the packets within a time window and provide information as to how much data was transmitted by each device within that period. Given a ground truth sensor, one can then identify causality between the ground truth sensor values and the patterns in the volume of data transmitted by each device in the space. In contrast to machine learning techniques, a causality approach allows SnoopDog to find the cause-effect relationship of arbitrary modality across any device that is transmitting causal data. One such method to find this cause-effect relationship is Granger Causality.

Granger Causality. A popular method to study causal relationships between two series is Granger Causality [9]. According to Granger Causality, if a series X Granger-causes series Y, then past values of X should contain information that helps predict Y above and beyond the information contained in past values of Y alone. Formally, if we have a series Y as:

$$y_t = a_0 + a_1 * y_{t-1} + a_2 * y_{t-2} + ... + a_n * y_{t-n}, \tag{4.1}$$

and we augment this series with the series X as follows:

$$y_t = a_0 + a_1 * y_{t-1} + ... + a_n * y_{t-n} + b_1 * x_{t-1} + ... + b_m * x_{t-m}, \tag{4.2}$$

then X Granger-causes Y if and only if Equation 4.2 gives a better prediction of y_t than Equation 4.1. Here, y_{t-k} are called lags of y and x_{t-k} are called lags of x where $k \in [1, n]$.

4.3 Characterizing a Representative Set of Snooping Sensors

In order to choose a set of ground truth sensors that can capture causality across any modality, we focus on generalizing across a representative set, including cameras, RF, and arbitrary sensors that report inferred (as opposed to raw) events.

Visual sensors. Wireless cameras are typically encoded with a codec that recognizes underlying patterns in the frames of the video and utilizes this information for compression. One such codec is H.264 [44]. An encoder first encodes the video using the standard, and a decoder then reconstructs the original video with minor information loss.

Standard temporal compression algorithms compress the video with 3 key frame-types, denoted I, P, and B frames–as shown in Figure 4.1. I frames (Intra-coded picture) hold complete image information, whereas P and B frames contain fractional image information, i.e., scene differences. As I frames are a complete image, they do not require any other frames to be decoded. P frames (Predicted picture) only contain changes in the image from previous frames. The information in a P frame is combined with the information of the I frame preceding it to obtain the resulting image. B (Bi-directionally predicted pictures) frames can construct the image from either direction. They can be coded with changes from the I or P frames before them, changes from I and P frames after them, or interpolation between the I/P frames before and after them. B frames are most compressible, followed by P frames, and finally, I frames.

Hence, with increasing motion in the scene recorded by an IP camera, there will be an increase in the data that must be transmitted due to the increase in the number of P and B frames sent. Camera traffic will increase as the number of pixels being perturbed in the scene increases; similarly, traffic will decrease if the scene transitions to a stationary one. As such, if a human subject were to perform some motion in the scene, stop for enough time to let the camera video settle down, and then move again, it will result in a unique camera traffic pattern that corresponds to the user's motion. This cause-effect relationship between human motion and camera traffic can then be used to discover if a wireless IP camera is

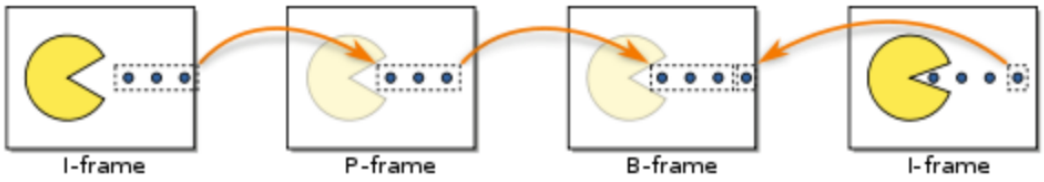


Figure 4.1: I-P-B Frames [41]

present in an occupied space. If there is no relationship between the camera traffic and user motion, then the camera is not monitoring the user.

RF sensors. Low cost, off-the-shelf millimeter-wave (mmWave) RF sensors are available that record the scene in the form of point-clouds. Recent works [34, 50] have shown that these point clouds can be used to infer human activity. However, unlike a camera, a radar device is a point scatterer, thus at any given time, only certain points in the scene reflect back. Hence, with motion in the scene, the number of points captured in every frame by the sensor (radar) vary considerably. In an empty scene, the number of points captured by these sensors is fairly constant but varies as subjects move about the space. The sensor also collects the velocity and intensity of the power received. This data helps the sensor in inferring fine-grained information about the space. If such a sensor live-streams point-cloud data over , the payload size will vary over time with changes in the number of points captured in the scene by the sensor. Hence, the network traffic will fluctuate with the number of points that are being captured in the frame. As such, there exists a cause-effect relationship between the subject's motion and the device's traffic.

Acoustic sensors. Another common type of bug used to snoop on people is microphones. With the growth in personal home assistant devices such as the Google Home or Amazon Alexa [18], it is trivial for someone to buy and install such listening devices in their homes. Although they are typically triggered by a keyphrase such as "Okay Google" or "Alexa", there are "Drop In" features that facilitate remote snooping. An adversary can also change the wake word of these devices to enable recording conversations of interest. Due to their compact form factor, they can be easily hidden. In such cases, this device will also work like an event-based clandestine sensor. Hence, services like SnoopDog that monitor traffic

for change in network patterns and either correlate them with another sensor recording of the same modality or find a cause-effect relationship with the ground-truth can detect their presence using network sniffing [17,45]. Here, instead of the IMU, we use the microphone on the user's smartphone as the trusted ground-truth sensor. In chapter 8-Q6, we discuss why it is challenging to detect and localize acoustic sensors that are continuously streaming.

Wireless sensors that encode inferred events. Motion sensors do not transmit a continuous stream of information. Most off-the-shelf motion sensors are passive infrared (PIR) based. They measure the infrared (IR) light from objects in their field of view. Any change in this incoming IR light is inferred as motion. Instead of continuously transmitting, they occasionally send data to their cloud service for processing once triggered by motion. Additionally, a camera can be programmed to continuously record video but only upload when a certain event occurs in the scene. These cameras behave like motion sensors and hence can be treated similarly. Virtual assistants also wait for trigger words to transmit a request to the associated cloud service, e.g., a user stating the device name to activate it [18].

Figure 6.7 shows the wireless traffic captured from an ordinary off-the-shelf motion sensor. Motion events in the scene trigger network activity. These events are a result of a subject moving in front of the device. Thus if a user moves around the room, stops, and moves again, there will be a unique cause-effect relationship between user motion and device traffic.

4.4 Device Identification via MAC Address

A MAC address is a universally unique ID assigned to the Network Interface Controller (NIC) for every networked device. It consists of 48 bits which are typically represented as 12 hexadecimal characters, i.e., `xx:xx:xx:xx:xx:xx`. The first 24 bits are the OUI (Organizationally Unique Identifier), which can uniquely identify a manufacturer or a vendor.

The MAC address of the sender and the receiver are contained within each exchanged packet. More importantly, this information is not encrypted. As a result, SnoopDog can easily obtain the MAC address to look up the device vendor. While we acknowledge

that the MAC address can be spoofed, this technique can still prove useful in the many cases where the adversary is a non-expert and thus has not spoofed the MAC. SnoopDog contains a database with names and MAC addresses of known vendors that manufacture surveillance devices. As SnoopDog detects more sensors, we add them to the available database[1]. Traffic fingerprinting techniques [2,5,8,25,26,30,51] can also be used to overcome the shortcomings of MAC-based identification.

[1]The link has been hidden in order to make the paper anonymous.

CHAPTER 5

Snooping Sensor Localization

Algorithm 1 details the **trial-based localization** used by SnoopDog to infer sensor location. In the case of multiple active sensors, this process can be repeated for each device.

Setup. Localization requires two input parameters: a region-of-interest to search over, and the snooping sensor's MAC address. To define the region-of-interest, we leverage Dead Reckoning [3, 21] for indoor user localization. For instance, a dead reckoning mobile application on a user's phone can instruct the user to walk the perimeter and capture the region boundary. Aside from identifying granger causality in traffic patterns, the MAC address is also used to ensure an appropriate trial method for localization (e.g., via techniques discussed in chapter 4.4 and [14]).

5.1 Identifying Sensor Coverage

Although the malicious sensor is known to monitor somewhere within the region-of-interest, it is unlikely to cover the entire region. Lines (1)-(8) narrow down the full search space into a bounding box *BBox* of the sensor's field-of-view. To begin, a user is instructed to traverse the region (line 2). At regular time intervals, the user's location is captured, and the snooping sensor's traffic is monitored for causality. Using the Granger Causality technique described in chapter 4, a particular location is identified as either within or outside sensor coverage. This process continues until the bounding box is determined to have sufficient density for performing trial-based localization, depending on the coverage area size.

The remainder of Algorithm 1 (lines 9-18) reduces the *BBox* scope of sensor coverage via directional elimination. Repeated trials are performed to specifically target high-probability

Algorithm 1: Localize identifies the location of a particular snooping sensor in a defined region-of-interest

Input: The sensor's MAC address

The $region$ of interest

Output: The sensor's location within the region

```
1  BBox ← ∅
2  traversing ← BeginTraversingRegion(region)
3  while traversing do
4      userloc ← DeadReckoningLocation()
5      inView ← GrangerCausality(MAC)
6      if inView then
7          BBox ← BBox ∪ {userloc}
8      traversing ← SparseBBox(BBox)
9  Loop
10     MLE ← MostLikelySensorLocation(region, BBox)
11     if SufficientBBox(region, BBox) then
12         return (BBox, MLE)
13     trialRegion = GenerateTrial(MLE, BBox)
14     inView = PerformTrial(trialRegion)
15     if inView then
16         BBox ← trialRegion
17     else
18         BBox ← BBox \ trialRegion
```

origins in order to either identify or eliminate likely sensor locations. Each round begins by solving for the most likely origin *MLE* for the sensor (line 10). While this process could be performed randomly, utilizing physical information about the current bounding box can significantly reduce the number of necessary trial rounds. For example, if the bounding box shape can be reasonably fitted to a triangle, then the sensor is likely horizontal-facing and placed on a wall. On the other hand, an ellipsoid coverage area likely indicates a sensor placed on the ceiling or floor.

An iterative process then proceeds to reduce the area of possible sensor locations to a pre-defined threshold (e.g., 10% of the region), upon which the bounding box and MLE are returned (line 11). In each iteration, a *directional* trial is conducted. **GenerateTrial** identifies a suitable position and heading for the trial by selecting a point near the center of the bounding box and facing the MPE (line 12). In our evaluation, we found distances of approximately 3 meters to be the maximum applicable distance for a trial. The trial takes one of many forms; for an inertial sensor, a user faces the designated direction and waves an object (e.g., hand or shoe) closely in front of their chest while shielding this activity with their body from any sensor present behind them. To trigger a camera sensor, a laptop plays a video clip that randomly flashes the screen with different colors. For audio, a trigger sound is played, and so on. If the trial results increased the device traffic, the bounding box is reduced to areas within visible range (line 16); otherwise, those areas are removed (line 18), and the next iteration begins.

5.2 Ensuring Sufficiently Reduced Region

In order to provide a guarantee that this localization method will always result in a minimal bounding box that is sufficiently small (e.g., 10% of the search region), a key assumption must be made: for any arbitrary bounding box, a trial can be identified which will eliminate a proper subset of the bounding box. In the case of Algorithm 1, this assumption can be reformed such that one can always construct a trial that eliminates at least a *single* point contained within the bounding box set. Due to the directional nature of each trial, this

can be achieved simply by conducting a trial that is positioned directly between two points within the bounding box, and facing directly towards one of the two points such that the other is obstructed. In the case of two points with large intermediate distances, a two-phase trial must be performed facing towards (and away from) each point, respectively.

Given the assumption that every trial can eliminate at least a single point from the bounding box set, guaranteeing that Algorithm 1 will always reduce the region to a certain size is trivial. In the worst case, for a bounding box of n points, *n-1* trials must be performed. In practice, each trial can eliminate many points contained within the bounding box. Furthermore, by leveraging the most likely sensor location, one can reduce the search space significantly and with relatively few trials.

CHAPTER 6

Implementation

This chapter presents an overview of our SnoopDog framework implementation by instrumenting readily available tools that are likely to be in a user's possession. We rely upon the following commonplace hardware and software.

6.1 Experimental Setup

Packet Sniffing: A laptop (Lenovo Thinkpad) is used to run the Wireshark network sniffing utility. The laptop's network card enters monitor mode and begins capturing all transmitted packets in the frequency band to aggregate traffic statistics for analysis. As it is not necessary to connect to a specific wireless network to monitor traffic, SnoopDog can capture and identify clandestine wireless sensors across all traffic, even if they reside on a closed or hidden network. A smartphone can also be used instead of a laptop, but requires a rooted [39] phone.

Collecting User's Motion Data: User's motion data is collected via the IMU present on the smartphone (Google Pixel 3). The smartphone is placed either in the user's hand or inside the user's pocket. 50 Hz accelerometer data is collected and used to study the cause-effect relationship between motion and sensor traffic. We collect along each of the 3 axes and use them separately as if motion is present in only one direction, the other 2 axes contribute minimally to the analysis, and may instead serve as noise. The smartphone is also used to collect audio and localize the user in his/her surroundings, which aids in localization.

6.2 Overview

SnoopDog sniffs the wireless traffic and aggregates the statistics over time, while user motion is captured using the IMU. This data is used to detect hidden sensors monitoring the user by measuring the cause-effect relationship between user motion and device traffic. SnoopDog also captures device MAC addresses to infer the manufacturer via an available database we have created. After detection, the trial-based algorithm is used to localize these sensors.

6.3 Aggregation of Traffic Statistics

Each device's traffic is grouped by MAC address, windowed, and processed to compute device traffic volume and variation. SnoopDog monitors packet sequence number in the WLAN layer to isolate and remove duplicate or redundant packets. As large images are sent over multiple fixed-length packets, a sufficiently large window size must be used. We chose a 100 ms window to group all packets with the same image within one interval. Cameras require a frame rate higher than 10 Hz to satisfy the flicker fusion (i.e., persistence of vision) threshold of the human eye [10, 23].

For camera encodings, we discard I-frames, as they do not encode differences in a scene and require higher bandwidth, thereby adversely affecting the causality analysis. To discover these frames, we first identify the camera frame rate by converting the time domain traffic to the frequency domain using a Fast Fourier Transform (FFT). The frame rate is the peak of the FFT, as shown in Figure 6.1. We then change the aggregation window size to correspond to this frame rate, calculate the data rate of the camera, as shown in Figure 6.3, and smooth variations with sliding window aggregates.

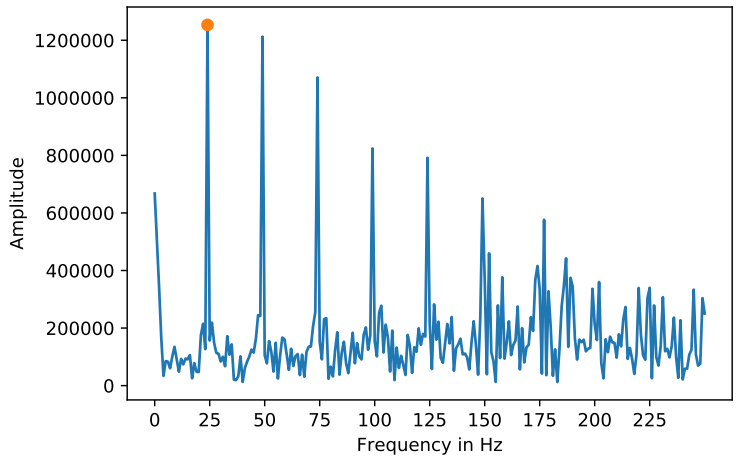


Figure 6.1: Detecting frame rate of the camera. In this case, the frame rate of the camera is 25 Hz which is where the peak is.

6.4 Detecting the Cause-Effect Relationship between User Motion and Hidden Devices

While sniffing the network, SnoopDog classifies the networked devices present into two categories: devices that transmit data continuously, and devices that have periodic or event-based transmission.

6.4.1 Wireless Sensors that Encode Raw Data

Some representative sensors that continuously transmit variably encoded raw data include camera and RF sensors.

Camera: When a camera is monitoring a static scene, the traffic is fairly constant, as shown in Figure 6.2. As the scene is perturbed by human motion, the wireless traffic changes rapidly. However, it is yet unclear whether human motion causes this variation. As soon as the user enters a new space, he or she can turn on , which works in the background to correlate IMU data with traffic of the transmitting devices. As users walk in a space, the starting and stopping patterns of their motion are unique. This unique pattern creates a fingerprint for the camera traffic. Once SnoopDog is able to determine a cause-effect relationship between device traffic and user's motion, it alerts the user. To definitively ascertain the presence of a

camera, SnoopDog enters phase two, where the user is asked to perform a stop-start-stop-start-stop (**S5**) motion as follows: 1) the user stays stationary for some time to allow the device traffic to stabilize. 2) The user performs jumping jacks at the current position. 3) The user stops again and waits for the camera traffic to settle. 4) The user performs jumping jacks again. 5) The user stops. This motions causes a pattern to appear in the traffic as shown in figure 6.3.

The entire detection phase requires $35 - 45$ seconds. While the user is performing the above **S5** motion, SnoopDog sniffs the packets on the network and records the user's IMU acceleration. Figure 6.3 plots the camera traffic after I-frame suppression and user accelerometer data while performing the **S5** motion. We observe that camera traffic is a function of human motion. When the human is static, the traffic is small, but when the human begins performing jumping jacks, the traffic rate increases. To prove that the accelerometer series indeed has an effect on the camera traffic, we leverage Granger Causality using the `statsmodel` package in Python. The null hypothesis of the Granger Causality Test is that the IMU series does not granger-causes the camera traffic series. Hence, if the p-value of our test is below a certain threshold of 0.08, we can reject the null hypothesis and claim that the IMU series granger-causes the camera traffic series.

RF sensor: the detection process remains the same for RF as that of a camera. We use an off-the-shelf mmWave RF sensor from Texas Instruments, as shown in [34]. We model the information obtained from the sensor as traffic. The modeled traffic from the RF sensor due to human motion is shown in Figure 6.4. Unlike a camera, RF sensors respond to either motion or other sources of RF in the space.

As soon as motion occurs within the space, the traffic changes rapidly in response. This is because the points captured by the RF sensor vary with motion. If the traffic of some device which was static when there was no motion but changes rapidly when there is motion and goes back to being static when motion stops, it is a clear indicator that the device is monitoring user movement. To detect such devices, SnoopDog first monitors the traffic when the scene is static. It then asks the user to perform the **S5** motion in the space while

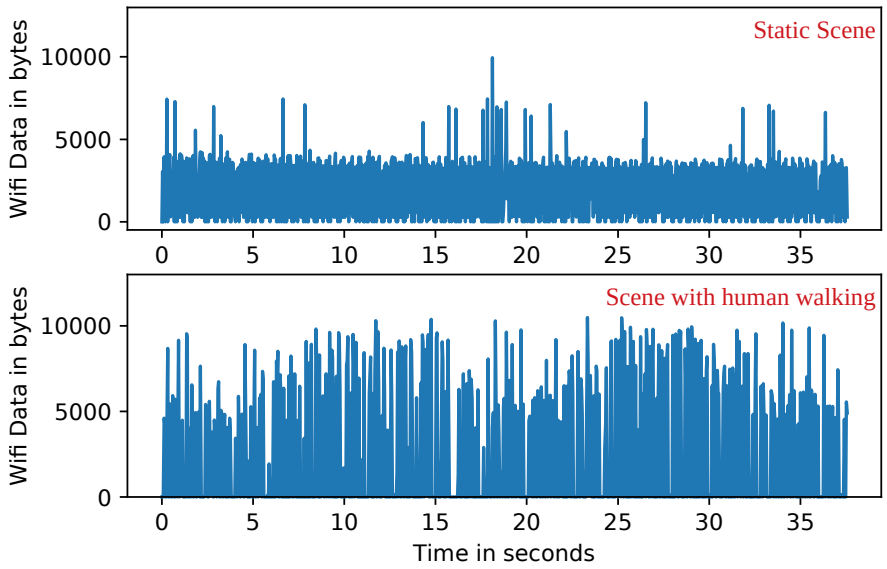


Figure 6.2: traffic captured from a camera over a static scene and a scene where a human is walking around.

SnoopDog monitors the traffic. As soon as the user is finished, the user should leave the space so that SnoopDog can monitor the traffic again and conclude the presence or absence of an RF sensor.

6.4.2 Wireless Sensors that Encode Inferred Events

Sensors that encode inferred events may transmit information periodically or upon event detection. By simply examining network traffic, it is difficult to ascertain if the device is transmitting periodic data, like a temperature sensor, or transmitting inferred events like a motion sensor.

Motion Sensor: Typical off-the-shelf motion sensors have a timeout to prevent continuous alerts. After the sensor detects a motion event, it stops inferring motion events for some time. If a human walks into the room, the motion sensor sends that information to a cloud server, which in turn sends an alert to the snooping user's smartphone or performs an action like turning on lights. After sending an alert, the sensor waits for the timeout period before it looks for more events. Most motion sensors have a timeout period between 30 seconds and 3 minutes. Similarly, there can be other sensors in the scene that have a timeout period

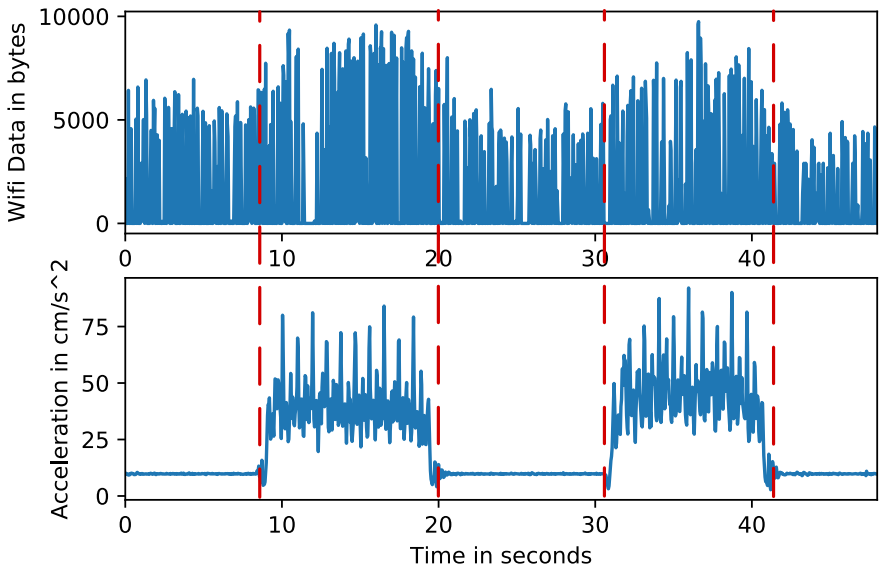


Figure 6.3: traffic of a camera and its comparison with IMU data of the user who was being monitored in the scene.

between uploading events. To discover a device's timeout period, SnoopDog correlates user movements with device traffic. If SnoopDog detects two events in the traffic series of a device and the user was in motion during the time between the two events, this time is noted as the timeout period. SnoopDog uses its active phase to further improve the timeout estimation by asking the user to move around the space until two events are detected in the device's network traffic. If the user wants even higher confidence in the detection, SnoopDog asks the user to move around the space, leave the space for the timeout period, and then move around the space again. After that, the user moves out from the space and then waits for the timeout period to end. If SnoopDog detects traffic by the device around the same time the user moved and none when the user is not moving, we can conclude that the traffic of the device is caused by user movement. This process can be repeated to increase the confidence of detection. A room can also be equipped with a camera that transmits motion events in the form of alerts. Such a camera can similarly be detected. In Figure 6.7, we move around the room (denoted by red dotted lines) and notice that the traffic from the motion sensor responds to these motion events.

Audio snooping: SnoopDog records user conversations in the background and correlates

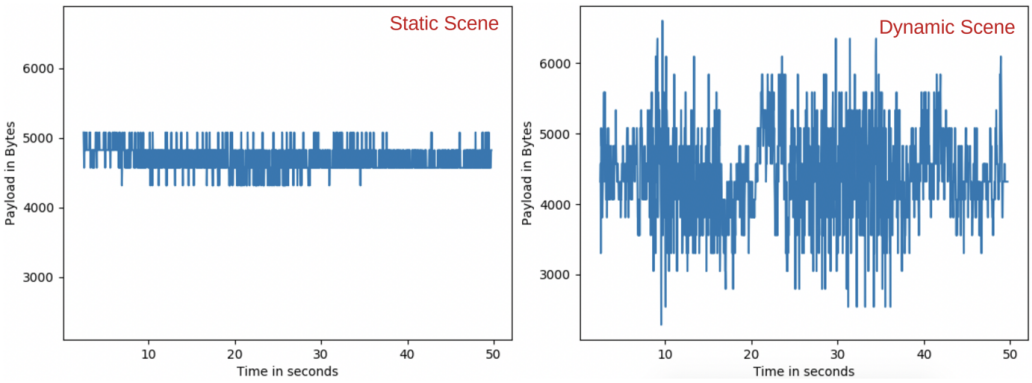


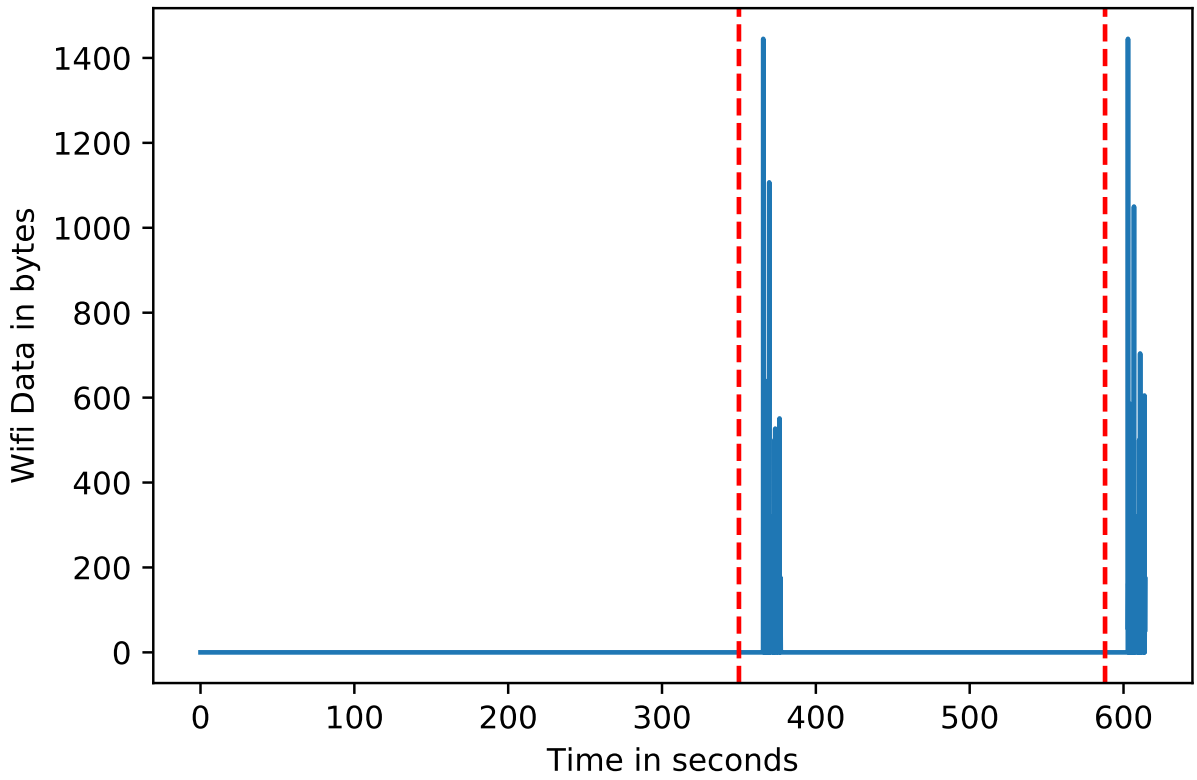
Figure 6.4: Modeled traffic for an RF sensor in a static scene and one where a user performs our detection trial.

it with the traffic of the devices on the network. If the occurrence of a certain phrase or a word cause the traffic of a device to change, SnoopDog asks the user to repeat those phrases until it can establish a cause-effect relationship between the occurrence of that phrase and the traffic of the device. Once SnoopDog knows the "wake word" for the acoustic home-assistant device, it repeats the recording several times while monitoring the device traffic to increase the confidence level of detection.

In our implementation, we used an Amazon Echo whose wake word was "Alexa". In Figure 6.7, we say the phrase "Alexa, what's the time right now?" four times and plot the device traffic. It is clear that these distinct peaks are a direct response to the trigger phrase. In 20 trials with different phrases, SnoopDog was able to detect causality 100% of the time.

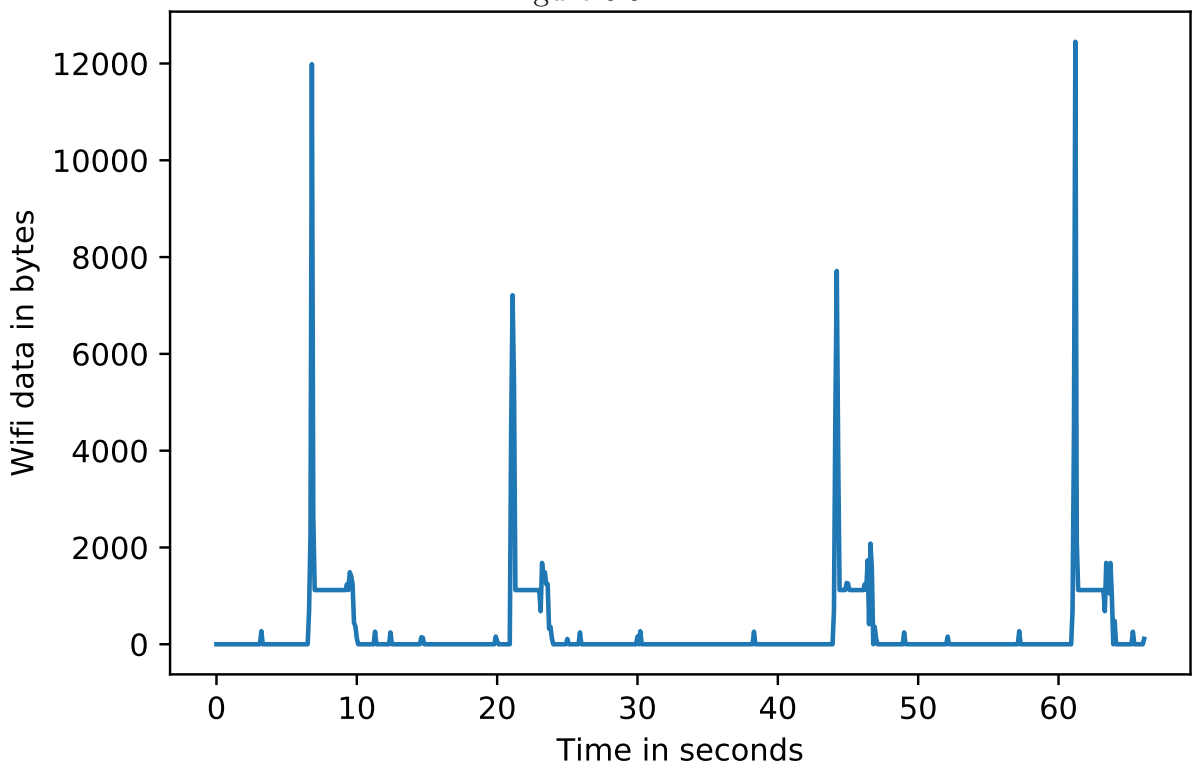
6.4.3 Device ID via MAC Address Lookup

SnoopDog checks its database for a match of OUI in the device's MAC address. If present, SnoopDog can inform the user with higher confidence that the device is indeed a surveillance device. Otherwise, it is added to the database and identified as a clandestine sensor.



.5

Figure 6.5:



.5

Figure 6.6:

Figure 6.7: (a) traffic of a motion sensor. The red-dotted line signifies a motion event. (b) traffic of an Alexa device for the user repeating the same phrase 4 times.

6.5 Device Localization

SnoopDog uses dead reckoning and asks the user to walk around the perimeter of the room to obtain a rough map of the room. Next, the user is asked to perform a detection trial at various locations within the room. More trials lead to better localization. At every location, SnoopDog tries to establish a cause-effect relationship with the device traffic. Regions with no cause-effect relationship are eliminated. This process is repeated to further reduce the search space for each clandestine device.

IP Camera: The traffic generated by a camera monitoring a scene will increase when the scene is dynamic. To exploit this, we first monitor the traffic of the device identified as a camera for 30 seconds over a static scene. Each trial consists of standing in a particular location (e.g., the middle of the scene), pointing a laptop in a particular direction, and playing a video that rapidly changes the colors on the screen of the laptop for 30 seconds. This process is then repeated in different directions. If the camera is able to monitor the laptop screen, its data rate during that period will be higher. On the other hand, if the laptop screen is not visible, the camera's traffic rate will be similar to the static scene. We can eliminate a fraction of the space where no activity is detected and repeat the process for the remaining region. In this way, we narrow down the possible region where a camera is located. We give a step by step walk-through of this process in chapter 7.

RF sensor: RF sensor localization is similar to that of a camera. However, since RF sensors cannot detect the flickering screen of the laptop, we use human movement. SnoopDog asks the user to stand in the middle of the space and wave their arm up and down rapidly in front of them while shielding this motion from the other side of the space with their back. If the RF device traffic does not respond to these stimuli when performed on one side but responds to it on the other side, we can eliminate that space.

Motion Sensor: Motion sensors are triggered by motion in front of them. SnoopDog first identifies the motion detector timeout (refer chapter 6.4.2), and then asks the user to stand in the middle of the room before the timeout expires. After timeout expiry, they are asked

to move their hand in front of them while shielding it from the other side with their body.

Acoustic (Audio) sensors: SnoopDog records the wake word of the device and asks the user to move around the room while this sound is repeatedly played from the smartphone app. If the user walks around the room but does not find any place where there the traffic of the device changes, we increase the volume and repeat the experiment. On the other hand, if the sound played at every point in the room causes the traffic of the device to vary, we decrease the volume and repeat the experiment. Finally, we identify areas where the sound causes network response and areas where it does not. We continue to reduce the volume of the device until the search space has been sufficiently reduced[1].

[1] A walk-through of this process is provided in chapter A of the Appendix.

CHAPTER 7

Evaluation

For evaluating SnoopDog , we used 4 cameras, 1 motion sensor, 1 Amazon Echo and 1 RF sensor. We selected off-the-shelf IP cameras at different price points to evaluate if we can achieve similar performance despite device heterogeneity.

7.1 Wireless sensors that encode raw data

Wireless IP Cameras. For Granger causality analysis, we lag the first series by one element at a time and observe what value of the lag results in the lowest p-value. Cameras have a delay between when the scene changes and when the data is visible to the adversary. We found that this delay can vary between a few milliseconds to up to 4 seconds. If the adversary is using a tape delay in transmission, we can perform this analysis over a longer delay period. In this time, the camera captures the video, encodes it, and sends it to its cloud server, which then forwards it to the receiving display. Assuming symmetrical delay, SnoopDog sniffs the packets during the first half of the transmission; we choose a lag value of 2 seconds.

The p-value threshold below which SnoopDog claims a successful detection is set at 0.08. We selected this using the results obtained from the first camera. However, we evaluate our detection with all the other cameras and show that this p-value threshold is optimal for all the cameras.

We evaluated our detection for 4 cameras – Foscam ($49.99), Kamtron ($39.99), Victure

Camera	Trials	Successful	Accuracy
Foscam	15	15	100%
Wansview	30	29	96.6%
Kamtron	25	21	84%
Victure	26	26	100%
Total	**96**	**91**	**94.7%**

Table 7.1: Evaluation results for camera detection

($35.99), and Wansview ($29.99). We performed 80 trials on 2 different users[1] to evaluate the detection accuracy. The results of our experiments are presented in table 7.1. To improve the detection accuracy and confidence of detection, a user can perform the detection trial several times and take a majority vote. The detection works well even when a portion of the human body is occluded by objects such as a table.

RF sensors. We use a TI mmWave IWR1443BOOST to evaluate the performance of SnoopDog for detecting RF sensors. We first monitored the traffic with no motion in the space and then asked the subject to move in and perform the detection trial. If the traffic of a device rapidly changes during movement but becomes stable if there is no activity, we conclude there is a cause-effect relationship between user motion and the device. In 20 experiments, SnoopDog was able to detect RF sensor presence every time.

7.2 Wireless sensors encoding inferred events

For sensors that encode inferred events, it is not possible to perform pure time-series Granger causality analysis to ascertain if there is a cause-effect relationship present between the sensor because their network traffic is discrete. Instead, we perform an activity and track network response. To detect the presence of an event-based sensor, we ask the user to move around

[1]The data is collected from the authors and hence does not require IRB approval.

the room, wait for the timeout period, and move around again. SnoopDog scans all device traffic within a period of 5 seconds after the motion to determine which device responds to user motion. If the device has traffic activity after the user moved, then the device is inferring events from the user motion. We evaluated this with an off-the-shelf motion sensor from Kangaroo Security. We performed 15 trials, and SnoopDog was able to detect this device every time.

7.3 Localization

We evaluated SnoopDog for 4 different spaces with different sensor placements. The accuracy of localization in all of these cases depends on the user's requirements. The user can perform more trials to reduce the probable region where the sensor is placed. We use an example to demonstrate how the SnoopDog localization algorithm works. To perform our localization, we chose a room as shown in Figure 7.1. The camera is placed at a corner of the room. We begin by performing our **S5** detection trials in different parts of the room. The location and results of our trials are shown. Based on these observations, we know that the camera is present somewhere in the square region of the room and hence, we eliminate the other part and start our trial-based localization.

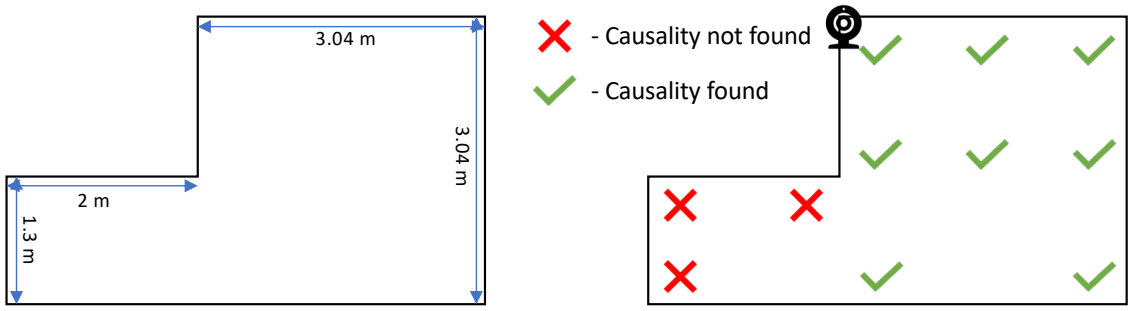


Figure 7.1: Lab dimensions and results of the detection trials.

We stand in the middle of the probable space and hold a laptop such that the screen is pointing in one direction. Then we turn to the other side and repeat the same experiment. We observe that there is a significant (>150%) increase in the camera data rate when the laptop is pointed towards the left side. When pointed to the right, the data rate remains

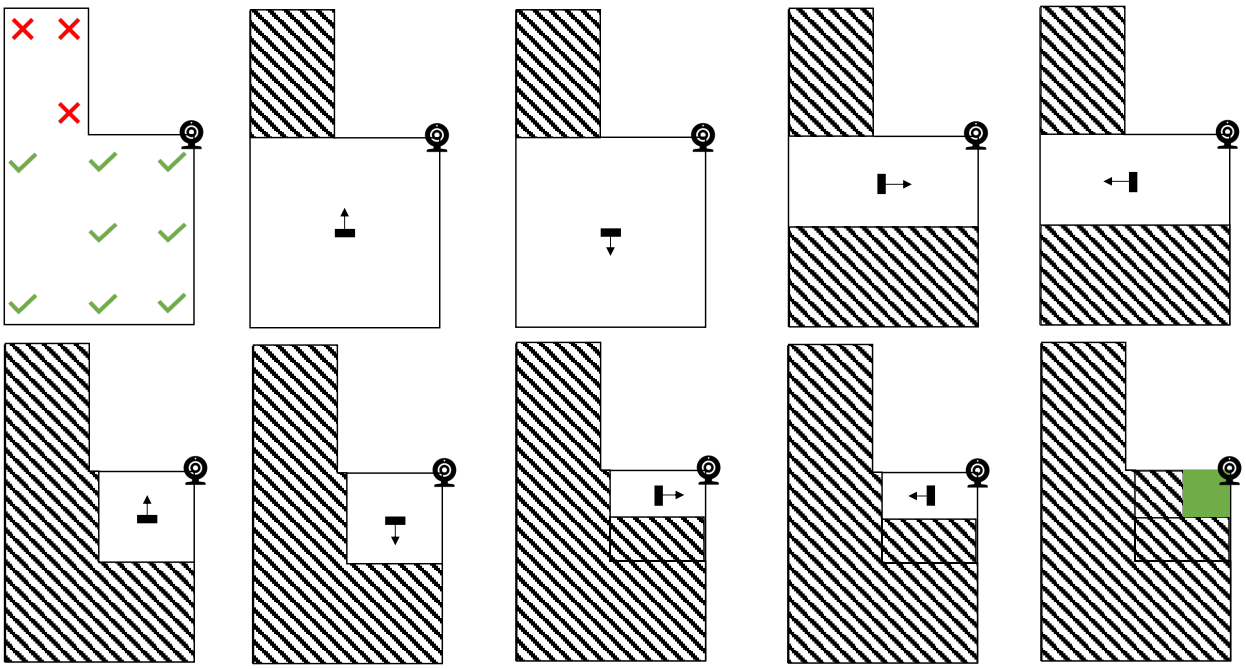


Figure 7.2: A walk-through of the trial-based localization algorithm in the laboratory environment in Figure 7.1. The arrows represent the direction the laptop screen was facing.

similar to that of an empty room. Thus we eliminate the right portion of the room from the probable area. We again stand in the middle of the leftover space and repeat the experiments until we achieve a sufficiently reduced space.

Audio-based localization: A similar elimination-based localization for audio sensors is described in Appendix A.

7.4 Overhead Analysis

Time: Sensor detection can happen in the background with minimal user intervention. However, this will take some time. In situations where a user wants to immediately know if he/she is being spied on by a sensor (such as when entering into a changing room), they can skip the first phase and directly begin the second phase where they will perform the **S5** motion. It takes about 40 seconds to perform active detection. For localization, each trial can take 30 seconds. Since the localization space reduction is determined by the user, he/she

can perform the trial any number of times. If the total number of trials is n, the overhead will be about $30n$ seconds.

User effort: If the detection occurs in the background, there is no overhead in terms of user involvement. However, both active and localization phases require user effort. In case the user is suffering from physical disabilities, he/she may find it hard to follow through these steps.

7.5 Effects of range on detection

The range of a device plays an important role in its detection. The camera range is the depth which it can record meaningful information from the scene. If the user stands too far from the camera, the **S5** trial may not produce enough variation in the camera traffic for the cause-effect relationship. For motion sensors and RF sensors, the range is specified by the vendor. The motion sensor that we used has a range of 15 feet (4.6 m) and the RF sensor we used had a range of 4 m. For Amazon Echo, the range is a function of the loudness of sound. For the camera, we perform our detection trials at different distances to find out what the range of detection is.

We perform the **S5** motion in front of a camera at 1 m and gradually increase the distance. We find that as we move away from the camera, the changes in its traffic decrease. For the Wansview camera, the distance at which traffic changes are not enough to establish causality is the smallest at about 3 m.

7.6 False negatives for background detection

Figure 7.3 shows camera traffic and IMU data for a user walking randomly in a room. As long as the user stops at some point within the detection period, SnoopDog is able to detect hidden devices. However, if the user is continuously stationary or continuously in motion, SnoopDog fails to detect the presence of clandestine sensors.

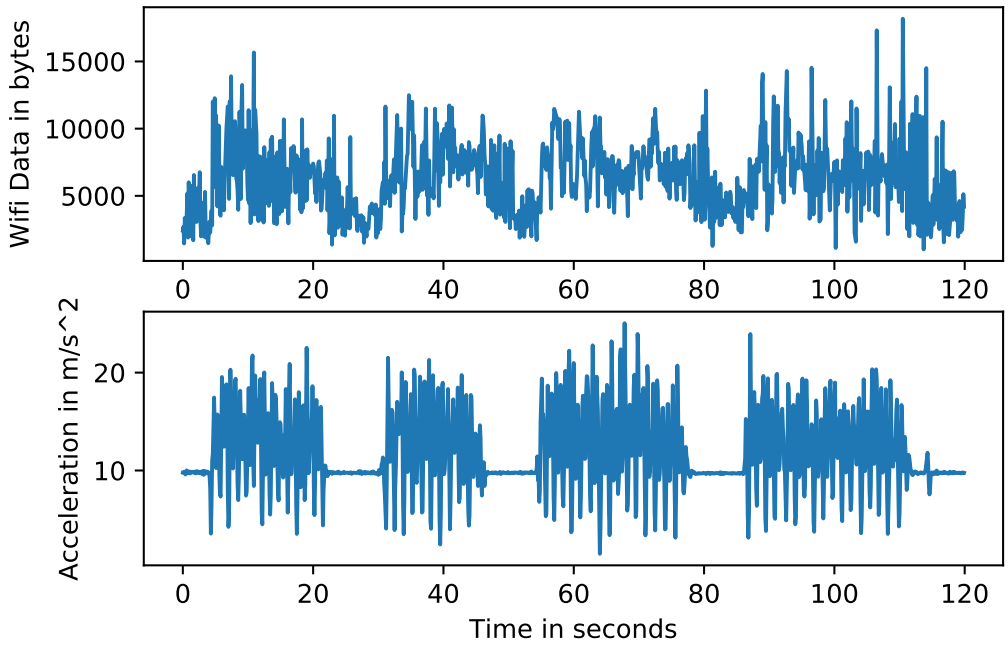


Figure 7.3: traffic of a camera and its comparison with IMU data of a user walking randomly with stops

CHAPTER 8

Discussion

Q1: What is the usability of ? We envision SnoopDog to be implemented as an app on either a smartphone or a smartwatch (or a combination of the two). This means an end-user will not need any prior knowledge about causality and coverage of a device to use it. SnoopDog will continuously work in the background to look for a cause-effect relationship between a user's actions and device traffic. It will then guide a user step-by-step through the entire localization procedure. Since an adversary can place a sensor at any time (e.g.,when a user checks in a room, searches for devices, finds none and then leaves for dinner after which the adversary places the spying device.), SnoopDog will still find it because it continuously works in the background. This will not cause any overhead in terms of user involvement.

Q2: Can* SnoopDog *detect any wireless sensor? Although SnoopDog can detect a wide variety of sensors, it cannot detect *any* wireless monitoring sensor. For a sensor to be detectable by , the traffic must be encoded with a Variable Bit Rate (VBR) algorithm and the data recorded by the sensor must change in response to user perturbation which can be recorded by a ground truth sensor. That said, most surveillance devices such as cameras, motion sensors and smart-home assistants today fall into this category, and thus we believe SnoopDog can serve as a valid defense.

Q3: How can false positives be reduced? For false positive to occur during active detection, the device's traffic needs to map directly to the **S5** motion during the active phase and user's motion during the background phase, which is unlikely. If there happens to be another camera in an adjacent space monitoring another user who is performing the detection trial within the same time window as the first user, it will trigger a false detection.

However, this probability is extremely low. We were unable to identify false positives over our network evaluation. Nevertheless, it remains a possibility, and mitigating such instances are highly desirable.

Simple strategies can significantly reduce the chances of false positives. First, during the initial monitoring phase for wireless devices, any periodic trends in traffic patterns can be noted; the detector trial should ensure its periods are not synchronous with such periodicity. Furthermore, the detection process can be done multiple times with varying and erratic period lengths. This will drastically decrease the chances of a false positive, as a device would have to coincidentally follow this effectively random traffic pattern. Finally, the entire process itself can be performed repeatedly; each iteration compounds the decrease in false positive rate, such that it eventually reduces to a statistical impossibility.

Q4: How can devices fool ? If the adversary suspects that the subject is using , they can either modify the encoding schemes, turn off the device, use data padding, add random noise, or vary the resolution of the data being transmitted. We detail these approaches in Appendix B. However, if the traffic of THE device changes drastically when the detection trial is performed, this in itself is a form of causation and SnoopDog can detect it.

Q5: Are there alternative approaches to causality? One alternative approach to detecting snooping sensors is correlation. Correlation measures the size and direction of the relationship between two variables. If the values of two variables change at the same time and in the same direction, they are highly correlated. However, correlation does not imply causation. If we have a sensor that measures the ground truth in the modality we want to detect, we need to use causality analysis. For example, it takes the camera some time to process the information and send it over to the server. So if we capture human motion with an IMU, the camera traffic will lag the IMU time series. This is correctly captured by causality analysis but not by correlation. However, if instead of using a sensor to measure the ground truth, we use another sensor that can capture the same modality that we are trying to detect, we can use correlation because if both the devices are capturing the same event, their traffic should show similar trends.

Future work can also explore the efficacy of data-driven approaches such as deep learning for time series classification.

Q6: Can* SnoopDog *work for other wireless communication standards like Bluetooth, Zigbee, and Z-Wave? Although SnoopDog targets -connected devices, we can generalize the same framework for other popular wireless communication standards. This framework can be extended to standards like Zigbee [19], Z-Wave [49], and Bluetooth [11,27] as long as we have the following: 1) A receiver that can scan their probable frequencies and sniff their packets to find if any devices are transmitting and 2) the ability to find unique device IDs from packet headers and distinguishing header information from payload size.

Q7: What happens when there are multiple people present? When there are multiple people present in the space, we need to ask everyone to leave during detection and localization. In cases where other users are non-cooperative, their motion will affect the network traffic of these devices and cause false alarms or false negatives.

Q8: Can we detect continuously streaming audio bugs? There are two ways to encode audio, either constant bit rate (CBR) or variable bit rate (VBR). VBR techniques make use of similarity in sound, such as prolonged silence, to reduce the amount of data required for encoding. In contrast, CBR always encodes with the same number of bits. Many off-the-shelf audio recorders and audio streaming apps use CBR. Since SnoopDog only has access to the payload size of a packet, there must be variation in the payload to determine causality. Hence, SnoopDog cannot detect CBR audio bugs.

CHAPTER 9

Related Work

This chapter presents the most relevant and related works.

Detecting hidden devices using RF signals. A popular tool to detect hidden devices is called a bug detector [29], which is an RF receiver that can sense if the received power in a particular frequency range is above a threshold. The problem with such devices is that they are not reliable, and can produce false alarms when used near other sources of RF signals such as mobile phones or laptops [32, 42]. Also, they give no additional information about the type of device and where it is located. After detection, the onus lies completely on the user to physically find the device and verify if it is a hidden surveillance device or not. The host may have a wireless device to monitor the power consumption of his property, but to the bug detector, it would seem similar to an IP camera.

Classifying devices on the network using wireless traffic sniffing. While services like Princeton IoT Inspector [14] collect traffic statistics to identify the types of devices present on the network, they fail to identify if those devices are indeed spying on the user or not. Just ascertaining the presence of a surveillance device is not enough. The device may be present outside the house or it may be monitoring some part of the house which was already disclosed by the home owner. In cases like this, just identifying such a device exists is not enough, we also need to determine two important facets – is the device spying on the user and is it located in an area of the house that has the potential to violate user privacy. Moreover, tools like this need to have access to the network in order to be effective. If the snooping devices are placed in a hidden network or on a password protected network, the use cases of such a tool are limited.

Other network traffic analysis tools [1, 33] utilize traffic data to find which devices are consuming high bandwidth. Such techniques can be used to classify audio and video data streams present in the wireless networks. However, with an increase in streaming services [20, 38], it is difficult to distinguish camera video and audio flows with those of streaming services based on just their bandwidth usage.

Detecting cameras on the network using wireless traffic sniffing. In [43], Wampler et. al. shows that information leakage occurs in camera traffic due to how videos are encoded. They observe that changing lighting conditions cause noticeable variations in the network traffic. Several works [23,28] leverage this observation to detect cameras monitoring an environment. Though these techniques perform well, their performance degrades when the environment lighting changes naturally. Additionally, while this technique works well for a camera, it does not generalize to other types of snooping devices, like RF sensors or motion detectors. Finally, in order to be able to change the lighting conditions of a space, the user requires either specialized hardware (like an LED board or a bulb) or access to lighting controls, which is not guaranteed.

Data driven approaches like DewiCam [4] extract features from the intrinsic camera traffic patterns to train a classifier which can detect cameras. They exploit the correlation between human motion and camera data flows to determine if the camera is indoors or outdoors. However, it is unclear if such an approach will hold true over diverse set of cameras with varying processing speeds and data flows.

In [47], Wu et. al. present another technique to detect hidden streaming cameras through simultaneous observation. The authors use their own camera to record a scene while simultaneously sniffing the network traffic. They compare the data rate and pattern of their trusted camera with other devices in the network to look for any similarities. If a similarity exists, there is a high probability that the device is a camera.

Localizing wireless devices using RSSI. Received Signal Strength Indicator (RSSI) is the estimate of the power received at the receiver from the transmitter. As the distance between the transmitter and the receiver increases, the power received drops, and so does

the RSSI. This property is leveraged to localize devices using RSSI [22, 24, 40, 48]. However, due to phenomenon like multipath and shadowing, the accuracy of RSSI based localization varies from space to space [16]. As a result, the error is very high (in order of several meters). For small rooms, such a result will be meaningless, as the snooping device can be effectively hidden anywhere.

9.1 Conclusion

In this paper, we presented SnoopDog , a framework to detect, identify, and localize any wireless sensor monitoring a person in an arbitrary space. SnoopDog works by establishing causality between a set of ground truth sensors monitoring a user and the transmitted information of wireless devices on a WiFi network. It then uses this causality to perform trial-based localization. We implement SnoopDog on a set of commonly available devices such as a smartphone and a laptop and evaluate our solution on a set of representative clandestine sensors. The framework had a detection rate of 96.6% and a device classification rate of 100% when the injected multi-modal event was human motion or sound.

Bibliography

[1] Monitor wi-fi traffic - wireless bandwidth monitoring tools.

[2] Noah Apthorpe, Danny Yuxing Huang, Dillon Reisman, Arvind Narayanan, and Nick Feamster. Keeping the smart home private with smart(er) iot traffic shaping, 2018.

[3] Stephane Beauregard and Harald Haas. Pedestrian dead reckoning: A basis for personal positioning. In *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, pages 27–35, 2006.

[4] Yushi Cheng, Xiaoyu Ji, Tianyang Lu, and Wenyuan Xu. Dewicam: Detecting hidden wireless cameras via smartphones. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pages 1–13. ACM, 2018.

[5] Manuel Crotti, Maurizio Dusi, Francesco Gringoli, and Luca Salgarelli. Traffic classification through simple statistical fingerprinting. *ACM SIGCOMM Computer Communication Review*, 37(1):5–16, 2007.

[6] Jonathan D Fuller, Benjamin W Ramsey, Mason J Rice, and John M Pecarina. Misuse-based detection of z-wave network attacks. *Computers & Security*, 64:44–58, 2017.

[7] Sidney Fussell. Airbnb has a hidden-camera problem, 2019.

[8] Ke Gao, Cherita Corbett, and Raheem Beyah. A passive approach to wireless device fingerprinting. In *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, pages 383–392. IEEE, 2010.

[9] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, pages 424–438, 1969.

[10] Daniel G Green. Sinusoidal flicker characteristics of the color-sensitive mechanisms of the eye. *Vision research*, 9(5):591–601, 1969.

[11] Jaap C Haartsen. Bluetooth radio system. *Wiley Encyclopedia of Telecommunications*, 2003.

[12] Brian Heater. Amazon upgrades its blink outdoor security camera with better battery, two-way talk – techcrunch, May 2019.

[13] Xueheng Hu, Lixing Song, Dirk Van Bruggen, and Aaron Striegel. Is there wifi yet?: How aggressive probe requests deteriorate energy and throughput. In *Proceedings of the 2015 Internet Measurement Conference*, pages 317–323. ACM, 2015.

[14] Danny Yuxing Huang, Noah Apthorpe, Gunes Acar, Frank Li, and Nick Feamster. Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale. *arXiv preprint arXiv:1909.09848*, 2019.

[15] Sophie Jeong and James Griffiths. Hundreds of south korean motel guests were secretly filmed and live-streamed online, Mar 2019.

[16] SR Jondhale, RS Deshpande, SM Walke, and AS Jondhale. Issues and challenges in rssi based target localization and tracking in wireless sensor networks. In *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, pages 594–598. IEEE, 2016.

[17] Sean Kennedy, Haipeng Li, Chenggang Wang, Hao Liu, Boyang Wang, and Wenhai Sun. I can hear your alexa: Voice command fingerprinting on smart home speakers. In *2019 IEEE Conference on Communications and Network Security (CNS)*, pages 232–240. IEEE, 2019.

[18] Veton Kepuska and Gamal Bohouta. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 99–103. IEEE, 2018.

[19] Patrick Kinney et al. Zigbee technology: Wireless control that simply works. In *Communications design conference*, volume 2, pages 1–7, 2003.

[20] Deepak Kumar, Kelly Shen, Benton Case, Deepali Garg, Galina Alperovich, Dmitry Kuznetsov, Rajarshi Gupta, and Zakir Durumeric. All things considered: an analysis of iot devices on home networks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1169–1185, 2019.

[21] Robert W Levi and Thomas Judd. Dead reckoning navigational system using accelerometer to measure foot impacts, December 10 1996. US Patent 5,583,776.

[22] Zhijing Li, Zhujun Xiao, Yanzi Zhu, Irene Pattarachanyakul, Ben Y Zhao, and Haitao Zheng. Adversarial localization against wireless cameras. In *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications*, pages 87–92. ACM, 2018.

[23] Tian Liu, Ziyu Liu, Jun Huang, Rui Tan, and Zhen Tan. Detecting wireless spy cameras via stimulating and probing. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 243–255. ACM, 2018.

[24] Xiaowei Luo, William J O'Brien, and Christine L Julien. Comparative evaluation of received signal-strength index (rssi) based indoor localization techniques for construction jobsites. *Advanced Engineering Informatics*, 25(2):355–363, 2011.

[25] Yair Meidan, Michael Bohadana, Asaf Shabtai, Juan David Guarnizo, Martín Ochoa, Nils Ole Tippenhauer, and Yuval Elovici. Profiliot: a machine learning approach for iot device identification based on network traffic analysis. In *Proceedings of the symposium on applied computing*, pages 506–509. ACM, 2017.

[26] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N Asokan, Ahmad-Reza Sadeghi, and Sasu Tarkoma. Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2177–2184. IEEE, 2017.

[27] Nathan J Muller. *Bluetooth demystified*, volume 1. McGraw-Hill New York, 2001.

[28] B Nassi, R Ben-Netanel, A Shamir, and Y Elovici. Drones' cryptanalysis-smashing cryptography with a flicker. In *IEEE Symposium on Security and Privacy (SP), Vol. 00*, pages 833–850, 2019.

[29] Nbc. How to detect hidden cameras, Aug 2019.

[30] Jorge Ortiz, Catherine Crawford, and Franck Le. Devicemien: network device behavior modeling for identifying unknown iot devices. In *Proceedings of the International Conference on Internet of Things Design and Implementation*, pages 106–117. ACM, 2019.

[31] Associated Press. Cops: Man secretly filmed dozens of women in changing room, Jan 2019.

[32] Dinesh Sathyamoorthy, Mohd Jalis Md Jelas, and Shalini Shafii. Wireless spy devices: A review of technologies and detection methods. *EDITORIAL BOARD*, page 130, 2014.

[33] Paul Schmitt, Francesco Bronzino, Renata Teixeira, Tithi Chattopadhyay, and Nick Feamster. Enhancing transparency: Internet video quality inference from network traffic. 2018.

[34] Akash Deep Singh, Sandeep Singh Sandha, Luis Garcia, and Mani Srivastava. Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar. In *Proceedings of the 3rd ACM Workshop on Millimeter-wave Networks and Sensing Systems*, pages 51–56. ACM, 2019.

[35] Cynthia Southworth, Jerry Finn, Shawndell Dawson, Cynthia Fraser, and Sarah Tucker. Intimate partner violence, technology, and stalking. *Violence against women*, 13(8):842–856, 2007.

[36] Inside Edition Staff. Couple says they found hidden camera pointing at their bed in carnival cruise room, Oct 2018.

[37] SSI Staff. Smart home devices market forecast to be growing globally at 31% annual clip, Oct 2018.

[38] Anne Steele. Music revenue surges on streaming subscription growth, Sep 2019.

[39] San-Tsai Sun, Andrea Cuadros, and Konstantin Beznosov. Android rooting: Methods, detection, and evasion. In *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, pages 3–14. ACM, 2015.

[40] Yuxiang Sun, Ming Liu, and Max Q-H Meng. Wifi signal strength-based robot indoor localization. In *2014 IEEE International Conference on Information and Automation (ICIA)*, pages 250–256. IEEE, 2014.

[41] Onur Uzun. I-p-b frames, Dec 2017.

[42] Veronica Valeros and Sebastian Garcia. Spy vs. spy: A modern study of microphone bugs operation and detection. Chaos Computer Club e.V., 2017. https://doi.org/10.5446/34936 $Lastaccessed : 26Nov2019$.

[43] Christopher Wampler, Selcuk Uluagac, and Raheem Beyah. Information leakage in encrypted ip video traffic. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7. IEEE, 2015.

[44] Stephan Wenger. H. 264/avc over ip. *IEEE transactions on circuits and systems for video technology*, 13(7):645–656, 2003.

[45] Charles V Wright, Lucas Ballard, Scott E Coull, Fabian Monrose, and Gerald M Masson. Spot me if you can: Uncovering spoken phrases in encrypted voip conversations. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 35–49. IEEE, 2008.

[46] Haitao Wu, Kun Tan, Jiangfeng Liu, and Yongguang Zhang. Footprint: cellular assisted wi-fi ap discovery on mobile phones for energy saving. In *Proceedings of the 4th ACM international workshop on Experimental evaluation and characterization*, pages 67–76. ACM, 2009.

[47] Kevin Wu and Brent Lagesse. Do you see what i see? detecting hidden streaming cameras through similarity of simultaneous observation. *arXiv preprint arXiv:1901.02818*, 2019.

[48] Weixing Xue, Weining Qiu, Xianghong Hua, and Kegen Yu. Improved wi-fi rssi measurement for indoor localization. *IEEE Sensors Journal*, 17(7):2224–2230, 2017.

[49] Muneer Bani Yassein, Wail Mardini, and Ashwaq Khalil. Smart homes automation using z-wave protocol. In *2016 International Conference on Engineering & MIS (ICEMIS)*, pages 1–6. IEEE, 2016.

[50] Renyuan Zhang and Siyang Cao. Real-time human motion behavior detection via cnn using mmwave radar. *IEEE Sensors Letters*, 3(2):1–4, 2018.

[51] Chaoshun Zuo, Haohuang Wen, Zhiqiang Lin, and Yinqian Zhang. Automatic fingerprinting of vulnerable ble iot devices with static uuids from mobile apps. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1469–1483. ACM, 2019.

APPENDIX A

Audio-based Localization for Personal Home Assistants

In this chapter, we describe the audio localization technique step-by-step. First, we find the optimal volume at which when the sound (a phrase containing the wake word of the device) is placed on some points causes the device traffic to change. Then we go around the room while SnoopDog repeats that sound continuously and checks them for causality with device traffic as shown in Figure A.1. Sound played at the points marked as green produces cause-effect relationship with the device traffic. We eliminate the region where we detect no causality. Next, we reduce the volume by 1 level and repeat our experiment in the left-over space till we are left with a region of desirable size.

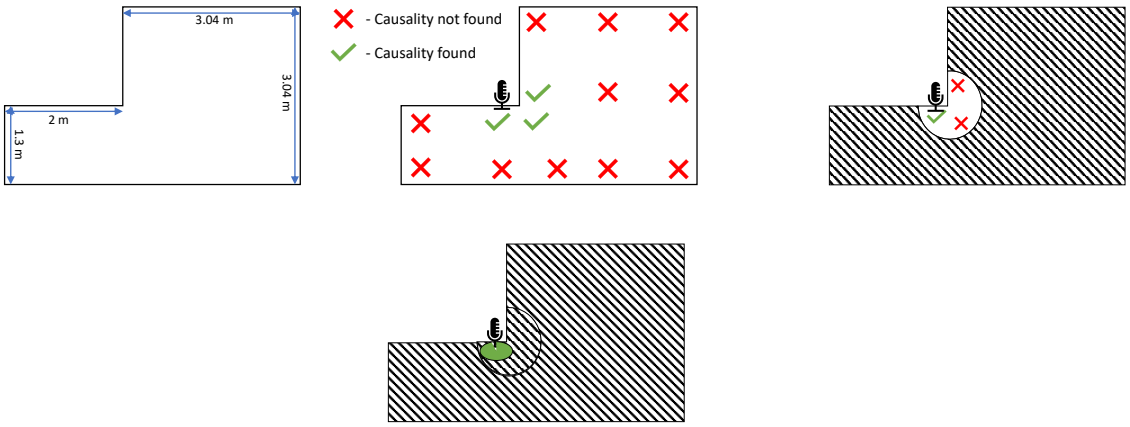


Figure A.1: Trial-based localization for acoustic sensors.

APPENDIX B

Techniques to fool SnoopDog

B.1 No Encoding or Data Padding

SnoopDog uses the relationship between encoding schemes and ground truth to find out if there is a device which is monitoring the user. Hence, to fool SnoopDog , the sensors can either send un-encoded raw data or they can pad the encoded data to make the data rate constant. Cameras can either pad their traffic or they can send un-encoded images frames. Since sending images will put a large overhead on the network bandwidth, padding the traffic [2] is a better idea. We pad the camera traffic wind random payload in Figure B.1. Since SnoopDog cannot see what's inside the payload, it can be anything. The device can even send labels in the payload that help the server decide if this is a valid packet or fake data generated to fool detection. Also in Figure B.1, we pad the traffic of a motion sensor to make it appear like a constantly transmitting device with no variation in traffic in response to user's motion.

For RF sensors, one can find out the maximum number of points it can output and then always pad the information so that we are transmitting the maximum number of points allowed. These extra points could all be zeros which would make it easier to filter them out on the server side.

Since motion sensors only send information if certain events occur, they can pad their traffic when no event occurs. As a result, they will have constant traffic for which causality analysis is not possible.

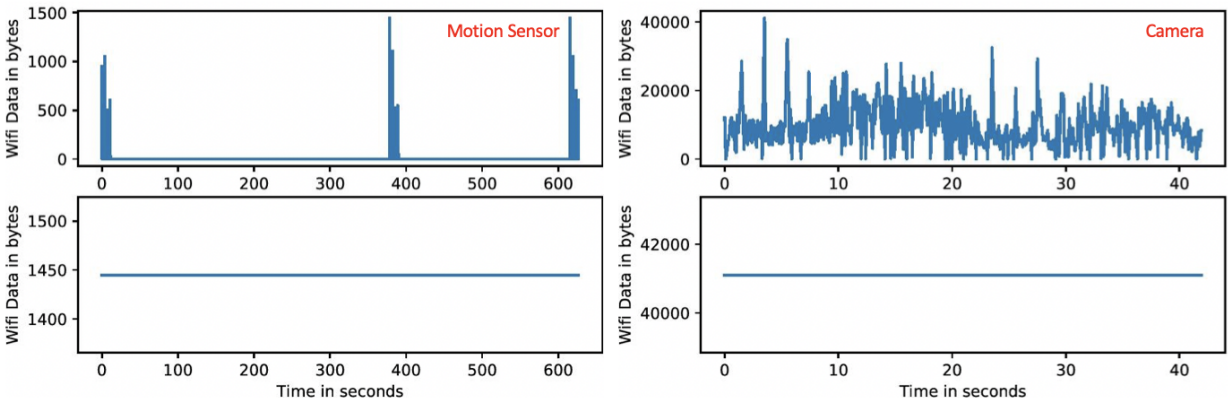


Figure B.1: Padding the motion sensor and the camera traffic

B.2 Adding Random Noise to the Data

Another way to fool SnoopDog is by injecting noise into the device's wireless traffic at random intervals for some time window. Since SnoopDog utilizes the change in device traffic to ascertain a cause-effect relationship, the variations caused by injecting random noise are able to fool the detection.

Devices that do not transmit continuously can randomly send information that creates a pattern similar to their inferred event traffic. This way they can keep sending their information which is hidden within random traffic. We add random noise which appears like regular traffic for a motion sensor in Figure B.2. This noise can be anything, and hence the server can differentiate it from actual motion events.

B.3 Constantly Vary the Resolution of the Data Being Transmitted

For devices like camera, there are several video resolutions that an adversary can choose. The higher the resolution, the better the video quality is. However, if an adversary chooses a scheme where the video resolution is constantly varying, it will cause random changes in the network traffic. Hence, even if the user's motion is causing changes to the traffic, it is

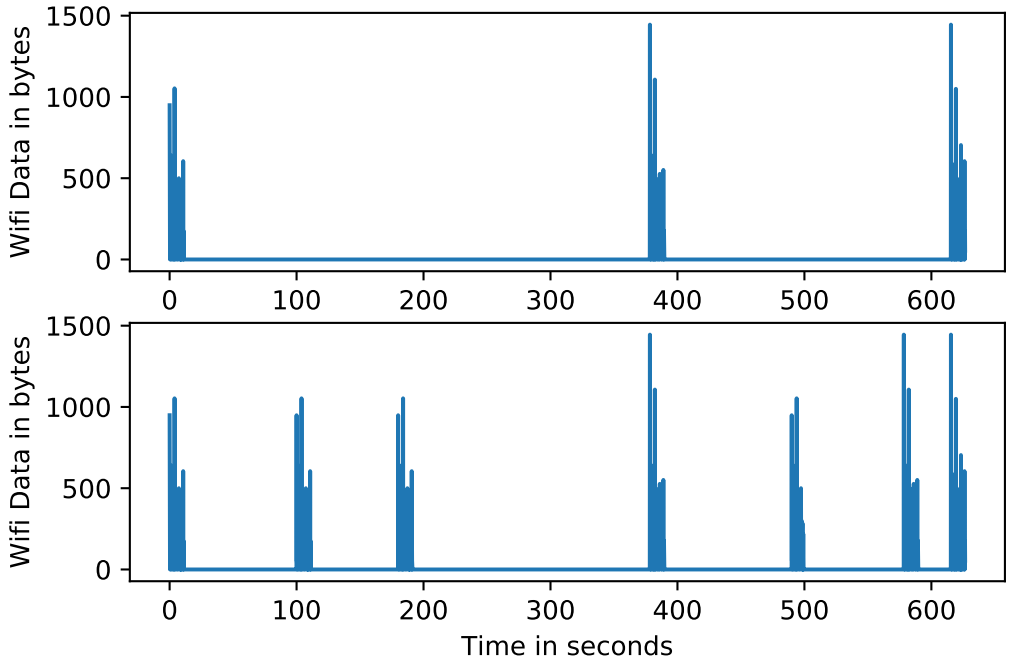


Figure B.2: Injecting noise in the traffic of a motion sensor to fool SnoopDog

overpowered by the changes in network traffic due to a variation in resolution.

For RF sensors, they can vary the number of maximum points that they transmit continuously to achieve a similar effect.

B.4 Adding a tape/broadcast delay to the transmissions

An adversary can add a tape delay to the sensor transmissions, i.e. intentionally adding a delay between when something was recorded and when it was transmitted. Since, we are only looking for causality within a small time window, a high tape delay will be able to fool SnoopDog . However, given enough storage capacity and time, it is possible for SnoopDog to scan the entire recording to look for cause-effect relationship with user motion. But for large tape delays, this is not practical.