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

Akash Deep Singh[†] , Luis Garcia[†,§] , Joseph Noor[†] , and Mani Srivastava[†]

akashdeepsingh@g.ucla.edu, lgarcia@isi.edu, jnoor@cs.ucla.edu, and *mbs@ucla.edu*
[†]University of California, Los Angeles (UCLA), [§] USC ISI (work done at [†])

Abstract

The increasing ubiquity of low-cost wireless sensors 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 generalized and comprehensive method to capture arbitrary sensors which may be "spying" on a user. In this work, we propose SNOOPDOG, a framework to not only detect common Wi-Fi based 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 for snooping devices 95.2% of the time, and localize devices to a sufficiently reduced sub-space.

1 Introduction

The proliferation of low-cost wireless sensors has facilitated increased adoption into smart home, building, and city deployments [1, 2]. 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.

An individual's privacy in these contexts is entirely at the discretion of the owner. Regulation is unclear in informal settings, such as a guest residing in a homestay lodging. There have been reported instances where a hosting owner has attempted to spy on homestay occupants [3], motel lodgings [4], and rooms aboard cruise ships [5]. There are even instances in well-established hotel chains and mall restrooms when a malicious employee or customer has bugged several rooms [6]. Beyond commercial applications, Southworth *et al.* report that domestic abusers may use such sensors for intimate partner stalking [7]. Thus, potential victims with privacy concerns must take a proactive approach to detect clandestine sensors.

The prevalent method to detect bugs relies on an RF receiver that senses if the received power in a particular frequency range is above a certain threshold. However, as bug detectors work on the principle of sensing surrounding RF signals, they can easily be triggered by legitimate RF devices such as mobile phones, radios, smart TVs, and other smart devices, thus limiting the practicality of these detectors. An alternate method has emerged to detect the presence of IoT devices based on network traffic statistics [8]. However, these methods only ascertain the presence of a device without semantic information regarding device information, location, or whether the device is actually monitoring a user.

More sophisticated solutions have since emerged targeting wireless cameras specifically. Wampler *et al.* [9] 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. Flickering a light source for a short period of time can then be used in correlation with network traffic changes to identify hidden cameras [10, 11]. Similarly, an approach has been presented that correlates the Wi-Fi traffic patterns of a trusted camera with Wi-Fi traffic patterns of other hidden cameras on a network to detect whether they are simultaneously observing the same space [12]. Unfortunately, these camera-specific approaches fail to generalize across modalities. For example, varying lighting conditions would be ineffective for detecting a hidden microphone or an RF sensor. In recent work, human motion was used to detect a hidden camera with coarse localization (i.e., indoors or outdoors) [13]. We argue that human motion is an emblematic event to generalize across modalities, as the objective in revealing bugs is typically to determine if the user is being observed.

In this paper, we propose SNOOPDOG, a generalized framework to detect clandestine wireless sensors 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 Wi-Fi traffic stemming from other devices. In particular, SNOOPDOG works by having the user perturb the trusted sensor values to observe if there is a causal pattern in the Wi-Fi traffic for a different device. For instance, if a wireless camera or a motion detector is monitoring a user who is wearing an inertial measurement unit (IMU), the IMU values will indicate a causal relationship with the camera's Wi-Fi traffic. SNOOPDOG utilizes encoding scheme models of different wireless sensing modalities to classify the sensor type, and then cross-references packet headers with publicly available information of manufacturers to identify the specific device model. We further introduce a novel fine-grained localization approach that leverages sensor coverage techniques to locate a detected sensor. We implemented SNOOPDOG using a user's mobile phone for ground truth sensors and a laptop for sniffing Wi-Fi traffic patterns. In the future, we envision SNOOPDOG to be implemented entirely as an app on either a smartwatch or a smartphone, both of which have sufficient sensing capabilities, but currently require Wi-Fi card improvements to allow for channel hopping in monitor mode, thus making SNOOPDOG easily accessible to non-technical users.

SNOOPDOG operates in two stages. SNOOPDOG begins in a *passive* monitoring phase that searches for suspicious causal patterns between the wireless traffic and the user's normal activity with their smartphone or wearable device. 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 accuracy. During the active phase, localization can optionally be engaged to find the clandestine sensor. The user can either skip the background or the active phase as 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 easily facilitates a user's search. This feature is a notable improvement over existing approaches that only localize devices as either indoors or outdoors. While SNOOPDOG cannot detect *any* wireless sensor monitoring the user (Section 9), it can detect a broad set of commonly used wireless sensors [14–16].

Contributions: Our contributions are summarized as follows:

- We propose SNOOPDOG, the first generalized framework to detect hidden clandestine sensors, including video, audio, motion, and RF. SNOOPDOG leverages the cause-effect relationship between a trusted set of sensor values and Wi-Fi traffic patterns when observing a multi-modal injected event.
- 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 reveal device information by cross-referencing packet inspection with publicly available device manufacturer information.
- We evaluate SNOOPDOG with a mobile phone and a Wi-Fi packet sniffer on a representative set of clandestine sensors and show a detection rate of 95.2% and device classification rate of 100% when the injected multi-modal event is human motion.

2 Background

We provide an overview of that state-of-the-art approaches to detecting the presence of wireless sensors in spaces. We then formalize the notion of detecting whether a sensor is monitoring a particular area.

2.1 Detecting Wireless Sensors in Spaces

The general approach to detecting wireless sensors relies on the notion that a device's wireless communication unintentionally leaks information in some out-of-band channel. Recent works exploited these leaks to detect the presence of wireless, transmitting bugs[1] in a space [17, 18]. 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 Wi-Fi, a device would scan frequency ranges around 2.4 GHz or 5 GHz. In tuning the received power threshold, there is a direct trade-off between detection accuracy and false positives [17]. 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 [18].

As wireless sensors transmit their information via packets, another technique to detect them uses packet sniffing. Approaches like DewiCam [13] 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 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

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

identify whether a device poses a potential privacy threat is to determine whether or not it is actively monitoring the user.

2.2 Detecting Sensors Monitoring a 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 variations due to motion within the captured scene–a characteristic exploited by prior research on detecting hidden cameras [10–12]. To generalize across sensor modalities, we formalize the notion that if an auxiliary sensor observes and measures a user's interaction with their surroundings, we can identify whether the user's actions indicate a causal relationship with the hidden sensor's wireless traffic. If such a relationship is found, then the sensor must be monitoring the user.

Detecting causality across sensor modalities. Given a target hidden sensor and access to its sensor data, we aim to establish causality between its time-series data and another sensor capturing the private space. A popular method to study causal relationships between two series is Granger Causality [19]. 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:

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

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

$$y_t = a_0 + a_1 * y_{t-1} + ... + a_n * y_{t-n} + b_1 * x_{t-1} + ... + b_m * x_{t-m}, \quad (2)$$

then X Granger-causes Y if and only if Equation 2 gives a better prediction of y_t than Equation 1. Here, y_{t-k} are called lags of y and x_{t-k} are called lags of x where $k \in [1,n]$.

In the following section, we discuss the system model and the design of SNOOPDOG.

3 SNOOPDOG Overview

We present the SNOOPDOG's threat model assumptions prior to enumerating the system design.

3.1 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 Wi-Fi patterns from the clandestine wireless sensor(s)[2]. These capabilities entail a set of assumptions.

Wi-Fi sniffing assumptions. We assume that the Wi-Fi sniffer on the user's device can monitor the encrypted traffic streaming from the clandestine device. SNOOPDOG does not require any 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 Wi-Fi encoding patterns of any clandestine wireless sensors. The notion of sufficient causality was formalized in Section 2.

3.2 Adversary Model

We focus on adversaries whose goal is to remotely spy on a third-party occupant of a private space in real-time. This model is consistent with other state-of-the-art methods for detecting hidden cameras [9–11, 13], and is supported anecdotally by several cases where owners were live-streaming guests in private spaces, e.g., [3, 4]. Further, many commercially available devices do not offer a local storage option for reasons of size, weight, power, and cost – such is the case with six out of the popular thirteen devices we examined. Moreover, live-streaming offers a more practical and scalable solution from a management perspective. Thus, 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 Wi-Fi, as this is the most prevalent method of wireless communication for remote monitoring using commercial and consumer equipment[3]. An adversary may use one of the several techniques mentioned in Section 8 to fool SNOOPDOG, for example with cover traffic or local storage. Implementing these techniques can require modifying the device firmware or physically interfacing with a proxy device (e.g., RPi), thereby increasing the barrier-to-entry for potential attackers. Moreover, techniques such as

[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.

[3] Although SNOOPDOG focuses on Wi-Fi-connected devices, we discuss in Section 9 how such a system could be generalized to other wireless communication standards and protocols.

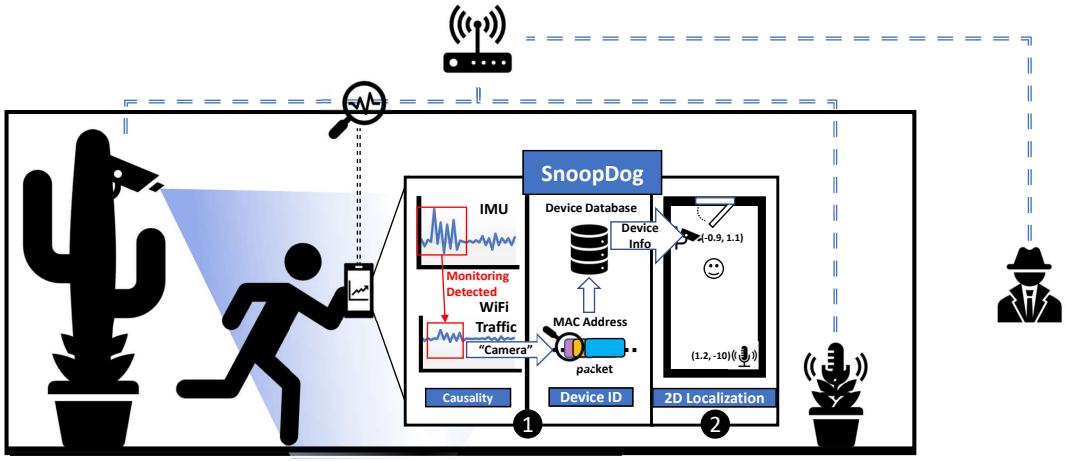


Figure 1: Overview of SnoopDog framework. ❶ The SNOOPDOG framework first identifies if a user is being monitored based on the cause-effect relationship between the values of a trusted sensor, e.g., an IMU, and Wi-Fi traffic patterns. It then inspects the associated packets and identifies the possible devices based on the physical (MAC) address. ❷ Finally, SNOOPDOG localizes each device by leveraging directionality and sensor coverage.

cover traffic can add significant and undesirable network overhead, particularly for a large number of sensors.

3.3 Design Overview

As depicted in Figure 1, SNOOPDOG detects and localizes a wireless sensor given 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 two phases. ❶ **Detecting and identifying snooping wireless sensors.** 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 Wi-Fi traffic patterns. If the user wants to scan 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 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. ❷ **Snooping sensor localization.** In the second 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.

4 Detecting and Identifying Snooping Wireless Sensors

This section 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 Wi-Fi network and connect the snooping device to it. As a result, the hidden device can be present in any of the possible Wi-Fi channels. Even though SNOOPDOG does not need access to these networks, it still needs to scan all Wi-Fi frequencies and look for any devices transmitting on them. 2.4 GHz and 5 GHz are the most popular bands for Wi-Fi 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 Wi-Fi Network Interface Card (NIC) scans through all channels sequentially to find available access points (APs) [20, 21]. Similarly, SNOOPDOG also scans through all the Wi-Fi 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 Wi-Fi channel. Thus SNOOPDOG can detect devices on any Wi-Fi network. Because devices may transmit data intermittently, SNOOPDOG continuously scans all Wi-Fi 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. Because we are interested in the causality between two sensors, SNOOPDOG will utilize Granger Causality (described in Section 2).

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 [22]. 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. 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 using either 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 traffic 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 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 [23, 24] 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. If such a sensor live-streams point-cloud data over Wi-Fi, 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 a microphone. With the growth in personal home assistant devices such as the Google Home or Amazon Echo (Alexa) [25], 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, these devices will also work like event-based clandestine sensors. 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 [26, 27]. Here, instead of the IMU, we use the microphone on the user's smartphone as the trusted ground-truth sensor. In section 10-Q4, 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 send data to their cloud service for processing once triggered by motion. 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. 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 uttering the device name to activate it [25].

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 Wi-Fi packet. More importantly, this information is not encrypted. As a result, SNOOPDOG can 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 address. Traffic fingerprinting techniques [28–34] can also be used to overcome the shortcomings of MAC-based identification. Additionally, in case of MAC randomization or MAC spoofing, techniques such as the ones described in [35] can be used to first track the traffic from a particular device and then perform cause-effect analysis on it.

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 database.

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 [36–38] for indoor user localization. A dead reckoning mobile application [36] on a user's phone instructs 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 Section 4.4 and [8]).

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
```

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 Section 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 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.

6 Implementation

This section presents the implementation details of SNOOPDOG. We use readily available tools that are likely to be in a user's possession.

6.1 Experimental Setup

Wi-Fi Packet Sniffing: The laptop's (Lenovo Thinkpad) network card enters monitor mode and uses Wireshark to capture all transmitted packets in the Wi-Fi frequency band to aggregate traffic statistics for analysis. As it is not necessary to connect to a specific Wi-Fi network to monitor traffic, SNOOPDOG can capture and identify clandestine wireless sensors across all Wi-Fi 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 data 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.

6.2 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.

6.2.1 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, its traffic is fairly constant, as shown in Figure 2. As the scene is perturbed by human motion, the 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 SNOOPDOG, which works in the background to correlate IMU data with Wi-Fi 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 on 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 asks the user 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 device traffic to settle. 4) The user performs jumping jacks. 5) The user stops. The S5 motion causes a unique pattern to appear in the Wi-Fi traffic as shown in Figure 3 (Cam. 2).

The entire detection phase requires $35-45$ seconds. While the user is performing the above **S5** motion, SNOOPDOG sniffs the Wi-Fi packets on the network and records the user's IMU acceleration. Figure 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 the threshold of 0.08, we can reject the null hypothesis and claim that the IMU series granger-causes the camera traffic series. We selected this p-value using the results obtained from the first camera. However, we evaluate our detection for all the other cameras and show that this p-value threshold is optimal for all the cameras.

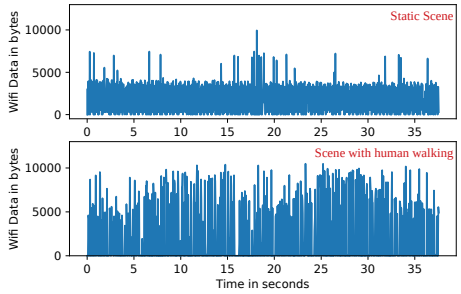


Figure 2: Wi-Fi traffic captured from a camera for a static scene and a scene where a human is walking around.

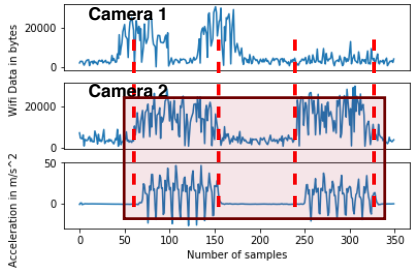


Figure 3: Wi-Fi traffic of a snooping camera placed in the same space as the user (Cam. 2) and a non-snooping camera placed in a different space (Cam. 1) and its comparison with IMU data of the user being monitored in the scene.

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 [23]. We model the information obtained from the sensor as Wi-Fi traffic. The modeled Wi-Fi traffic from the RF sensor due to human motion is shown in Figure 4. Unlike a camera, RF sensors respond to a change in RF reflections from the scene.

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 an 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 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.

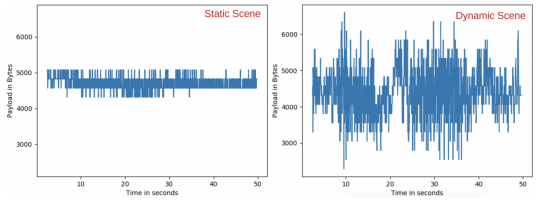
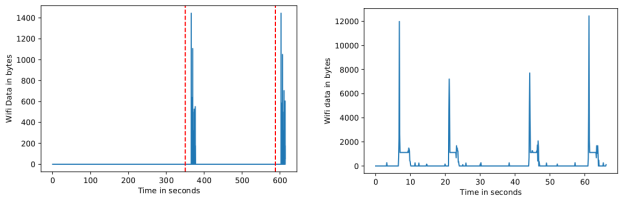


Figure 4: Modeled Wi-Fi traffic for an RF sensor in a static scene and one where a user performs our detection trial.

6.2.2 Wireless Sensors that Encode Inferred Events

Sensors that encode inferred events transmit upon event detection. By 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. The motion sensor sends motion 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. This period is between 30 seconds and 3 minutes for most motion sensors. Similarly, there can be other sensors in the scene that have a timeout period between uploading events. To discover a device's timeout period, SNOOPDOG correlates user movements with device traffic. If two events are detected in the traffic 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. 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, it concludes that the traffic of the device is caused by user movement. This process can be repeated to increase the confidence of detection. In Figure 5, we move around the room and notice that the Wi-Fi traffic from the motion



(a) (b)

Figure 5: (a) Wi-Fi traffic of a motion sensor. The red-dotted line represents a motion event. (b) Wi-Fi traffic of an Alexa device for the user repeating the same phrase 4 times.

sensor responds to these motion events. Since this traffic is discrete, we cannot perform time-series Granger causality analysis. Instead, we perform an activity and track network response. To detect the presence of a motion sensor, we ask the user to move around 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.

Audio snooping: SNOOPDOG records user conversations in the background and monitors the network traffic. If the occurrence of a certain phrase or a word causes the traffic of a device to change, SNOOPDOG asks the user to repeat it until it can establish a causality 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 and Echo Dot whose wake word was "Alexa" and "Computer" and a Google Home Mini with the wake phrase "Hey Google".

6.2.3 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.

6.3 Device Localization

SNOOPDOG uses dead reckoning [40] and asks the user to walk around the perimeter of the room to create rough map of the room. Next, the user performs a detection trial at various locations in 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.

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 section 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 section 6.2.2), 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[4].

7 Evaluation

We evaluated SNOOPDOG on a set of sensors from well-known brands as well as best-selling sensors on Amazon. These are listed below in Table 1.

7.1 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

[4] A walk-through of this process is provided in section A of the Appendix.

Name	Type	Cost
Kamtron	Camera	$39.99
Panasonic (HomeHawk)	Camera	$77.64
Wansview	Camera	$29.99
Arlo (NetGear)	Camera	$107.50
Victure	Camera	$35.99
Foscam	Camera	$49.99
Ring (Amazon)	Camera	$59.99
Amazon Echo Dot	Home Assistant	$29.99
Amazon Echo	Home Assistant	$99.99
Google Home Mini	Home Assistant	$39.99
Kangaroo Home	Motion Sensor	$12.95
Samsung Smart Things	Motion Sensor	$24.99
TI IWR1443	RF Sensor	$299.99

Table 1: List of snooping sensors evaluated upon

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. Assuming symmetrical delay, SNOOPDOG sniffs the packets during the first half of the transmission; we choose a lag value of 2 seconds.

We evaluated our detection on 7 cameras. All of them use H.264/MPEG-4 codecs which are the most popular codecs used for IP cameras. We performed 131 trials on 2 different users[5] to evaluate the detection accuracy. The results of our experiments are presented in table 2. 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.

Camera	Trials	Successful	Accuracy
Panasonic	15	14	93.33%
Arlo (Netgear)	10	10	100%
Ring (Amazon)	10	9	90%
Foscam	15	15	100%
Wansview	30	29	96.6%
Kamtron	25	21	84%
Victure	26	26	100%
Total	**131**	**124**	**94.65%**

Table 2: Evaluation results for camera detection

RF sensors. We use a TI mmWave IWR1443 to evaluate the performance of SNOOPDOG. In 20 experiments, SNOOPDOG was able to detect RF sensor's presence every time.

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

7.2 Sensors Encoding Inferred Events

Motion Sensors. We evaluated on an off-the-shelf motion sensor from Kangaroo Security and a smart-things motion sensor from Samsung. The smart-things sensors are a special case as these sensors use Z-Wave and ZigBee to communicate with a smart-things hub which in turn sends the information over Wi-Fi. As a result, SNOOPDOG can sniff the traffic of this hub and establish causality. However, if there are multiple devices connected to the same hub, SNOOPDOG will not be able to detect them. We performed 25 trials, and SNOOPDOG was able to detect the motion sensors every time except for 3 trials. We suspect that this was caused because the devices send some sort of "status" messages to their respective cloud service which result in events in the sniffed traffic that throw the detection off.

Smart-home Assistants (Audio Sensors). In Figure 5, we say the phrase "Alexa, what's the time right now?" four times and observe four distinct events in the device traffic. In 35 trials with different phrases, SNOOPDOG was able to detect causality 100% of the time. Additionally, we show the variations in device traffic for Alexa Drop-In features when the Drop-In is active and when it is not active in Appendix B.

7.3 Quantifying False Positives and Negatives

Overall false positive rate. We perform false positive analysis for two separate cases. In case one, we run cause-effect analysis for every device whose traffic has been captured (a network can have devices such as phones, laptops, and other sensors in addition to snooping devices) and find out if a device is falsely classified to be snooping on the user. In case two, we perform our cause-effect analysis on a camera that is not looking at the scene to see whether it will produce a false positive.

For case one, our experimental test-bed consists of devices monitoring the user such as a camera, motion sensor, home assistant and RF-sensor. We also include innocuous devices usually present in a scene such as smartphones, laptops, Wi-Fi-routers, smart-lights, and embedded devices such as raspberry-pi. We ran this analysis for 50 trials. We found 2 false positives for Active Phase and 9 false positives for Background Phase.

We divide case 2 into two separate parts. In the first part, a camera is placed in another room (where the user who's running SNOOPDOG is not present). Another person is moving freely in the room. In the second part, the person in the other room is sitting on a desk and working while the camera is monitoring them. We show the results of our experiments in Table 3. We perform these trial for both background detection and active detection. The one detection for background phase is because the walking patterns of the user and the person in the other room were similar. In such cases, a user can turn off the light in their room and see if the traffic of the device drops. If it does, it means the camera is in their room.

Case	Total Trials	False Positives	
		Background	Active(S5)
(1) User moving	15	1 (6.66%)	0 (0.00%)
(2) User working on a desk	10	0 (0.00%)	0 (0.00%)

Table 3: False-positives for a camera in a separate room.

False negatives for background detection. SNOOPDOG's background detection mechanism is most susceptible to false negatives since it relies on natural human motion rather than active command and response patterns. Figure 6 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.

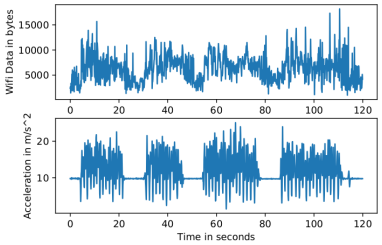


Figure 6: Camera traffic and its comparison with IMU data

7.4 Snooping Sensor 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. 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.

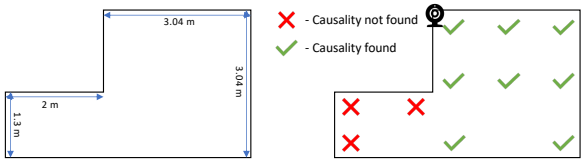


Figure 7: Lab dimensions and results of the detection trials.

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 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.

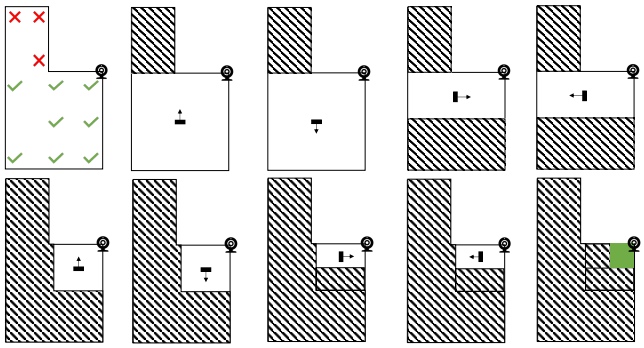


Figure 8: A walk-through of the trial-based localization algorithm in the laboratory environment in Figure 7. The arrows represent the direction the laptop screen was facing.

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

7.5 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 a changing room), they can directly begin the active 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 phase and localization require user effort. In case the user is suffering from physical disabilities, he/she may find it hard to follow through these steps.

Wi-Fi Channel Hopping: In detection mode, SNOOPDOG must hop channels across all frequencies as an adversary may have configured devices on different Wi-Fi frequency bands. A transmitting device might inadvertently evade detection if it is hopping while SNOOPDOG is scanning. Hence, we perform multiple scans of the entire Wi-Fi spectrum to reduce misdetections. We would ideally monitor all Wi-Fi channels simultaneously, but this is infeasible with a single WiFi radio. Multiple Wi-Fi radios will reduce the search space per radio at additional cost and degraded portability.

7.6 Effects of Range on Detection

Detection depends on the range of a sensor. The camera range is the depth which it can record meaningful information from the scene. If the user stands too far from the camera, their motion may not produce a significant variation in its traffic. 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 cameras, we perform our detection trials at different distances to find out the range.

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.

7.7 Failure Analysis

During our evaluation, SNOOPDOG's detection failed during some trials for both cameras and motion sensors. In this section, we will describe and discuss those cases.

Camera. SNOOPDOG failed to detect hidden cameras when the camera video froze (no transmission) for a small duration during our detection trial. Another cause of miss-detection were additional users in the scene since the network traffic due to their motion cannot be filtered out. Also, when the user was present far from the camera (refer Section 7.6), the traffic changes due to the motion of the user were not enough to establish Granger causality.

Motion Sensor. Motion sensors sometimes send status messages to the cloud service that they are connected to. These transmissions happen in the form of a short burst. Since there was no motion before these transmissions, they result in SNOOPDOG failing to establish causality. While we cannot say with certainty what each of these transmissions are, we saw multiple such transmissions when the sensor was started, during update, and sometimes even in between two motion events. A user can also query the sensor status from their phone to which the sensor has to respond. SNOOPDOG failed to detect causality when these transmissions were present in the network traffic of the motion sensor.

8 Techniques to fool SNOOPDOG

In this section, we discuss how an adversary can fool SNOOPDOG.

8.1 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 [30] is a better idea. We pad the camera traffic with random payload in Figure 9. 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 9, 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.

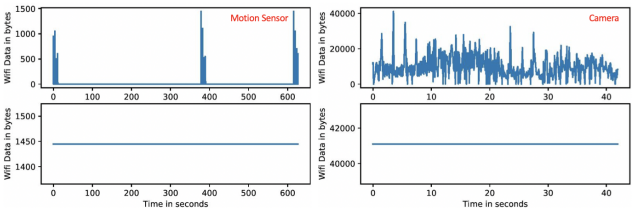


Figure 9: Padding the motion sensor and the camera traffic

8.2 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 10.

8.3 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 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

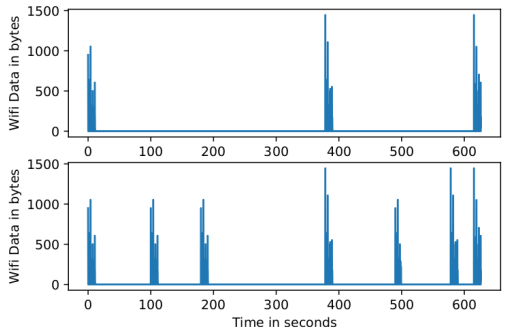


Figure 10: Injecting noise in the traffic of a motion sensor to fool SNOOPDOG

effect.

8.4 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.

9 Limitations

1: Only limited to VBR devices. Although SNOOPDOG can detect a wide variety of commonly available sensors, it cannot detect *any* wireless sensor monitoring the user. For a sensor to be detectable by SNOOPDOG, 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.
2: A technically capable adversary can fool* SNOOPDOG *if they know about its existence. If the adversary suspects SNOOPDOG is in use, they can use one of the techniques listed in Section 8. They can also use channel hopping or MAC randomization. We have not evaluated SNOOPDOG for any of the above techniques.
3: Evaluation is limited to Wi-Fi devices and devices who route their traffic through a Wi-Fi-hub only. We have evaluated SNOOPDOG for Wi-Fi-connected devices only. For future work, this framework can be evaluated for other popular wireless communication standards. SNOOPDOG can be extended to standards like Zigbee [41], Z-Wave [42], and Bluetooth [43, 44] 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. While capturing Zigbee/Z-Wave packets will require additional hardware, recent works have shown that it is possible for a Wi-Fi radio to perform cross-technology communication. [45, 46]

10 Discussion

Q1: What is the usability of* SNOOPDOG*? 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: 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, the probability of this happening is low. 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.
Q3: Are there alternative approaches to causality? One alternative approach to detecting snooping sensors is correlation. 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.

Q4: 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.

Q5: What is the impact of a ground-truth sensor? Qualitatively, the ground-truth sensor enables the detection of causality between human action and hidden sensors. Even if all hidden devices were connected to an accessible Wi-Fi network (which is the same system model used by IoTInspector [8]), one would only be able to detect the presence of a device on the network and not whether it is monitoring a user. To quantitatively demonstrate and evaluate the impact of a ground-truth sensor, Figure 3 illustrates an example where an IMU enables SNOOPDOG to identify between a hidden sensor monitoring a user and disregard a camera in a separate room. Moreover, one may argue that an application can actively instruct the user to move and establish causality between the period of instruction and the Wi-Fi traffic patterns. First, such an approach relies on a general user motion model to establish causality during these time frames. Second, this approach is not capable of background detection as it would rely on active command and response patterns. In Table 3 case 1, without a ground truth sensor, the false positive rate is 100%. With a ground truth sensor, this decreases to 6.66%.

11 Related Work

This section 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 [47] – an RF receiver that can sense if the received power in a frequency range is above a threshold. The problem with such devices is that they can produce false alarms when used near other RF sources such as mobile phones or laptops [17, 18]. Also, they give no additional information about the type of device or where it is located. After detection, the onus lies completely on the user to physically find the device and verify if it is a 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 [8] 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 [48, 49] 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 [50, 51], 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. Wampler *et al.* [9] and others [10, 11] show 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. Though these techniques perform well, their performance degrades when the environment lighting changes naturally. Additionally, while these techniques work well for a camera, they do 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.

Approaches like DewiCam [13] exploit the correlation between human motion and camera data flows to determine if the camera is indoors or outdoors.

In [12], Wu *et al.* use their own camera to record a scene while simultaneously sniffing the network traffic. They compare the data rate and pattern of their 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 transmitting device. The power received drops with distance, and so does the RSSI. This property is leveraged to localize devices using RSSI [52–55]. However, due to phenomenon like multipath and shadowing, the accuracy varies from space to space [56]. The error is very high (several meters). For small rooms, such a result will be meaningless, as the snooping device can be effectively hidden

anywhere.

12 Conclusion

In this paper, we presented SNOOPDOG, a framework to detect, identify, and localize Wi-Fi based sensors 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 Wi-Fi 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 95.2% when the injected multi-modal event was human motion or sound. SNOOPDOG leverages directionality of snooping sensors to reduce the total search area.

13 Acknowledgements

The research reported in this paper was sponsored in part by the National Science Foundation (NSF) under award #CNS-1705135, by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, and by the Army Research Laboratory (ARL) under Cooperative Agreement W911NF-17-2-0196. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ARL, DARPA, NSF, SRC, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] S. Staff, "Smart home devices market forecast to be growing globally at 31% annual clip," Oct 2018. [Online]. Available: https://www.securitysales.com/research/smart-home-devices-market-forecast/

[2] B. Heater, "Amazon upgrades its blink outdoor security camera with better battery, two-way talk – techcrunch," May 2019. [Online]. Available: https://techcrunch.com/2019/05/08/amazon-upgrades-its-blink-outdoor-security-camera-with-better-battery-two-way-talk/

[3] S. Fussell. (2019) Airbnb has a hidden-camera problem. [Online]. Available: https://www.theatlantic.com/technology/archive/2019/03/what-happens-when-you-find-cameras-your-airbnb/585007/

[4] S. Jeong and J. Griffiths, "Hundreds of south korean motel guests were secretly filmed and live-streamed online," Mar 2019. [Online]. Available: https://www.cnn.com/2019/03/20/asia/south-korea-hotel-spy-cam-intl/index.html

[5] I. E. Staff, "Couple says they found hidden camera pointing at their bed in carnival cruise room," Oct 2018. [Online]. Available: https://www.insideedition.com/couple-says-they-found-hidden-camera-pointing-their-bed-carnival-cruise-room-47948

[6] A. Press, "Cops: Man secretly filmed dozens of women in changing room," Jan 2019. [Online]. Available: https://www.wptv.com/news/world/police-more-than-60-victims-in-changing-room-camera-case

[7] C. Southworth, J. Finn, S. Dawson, C. Fraser, and S. Tucker, "Intimate partner violence, technology, and stalking," *Violence against women*, vol. 13, no. 8, pp. 842–856, 2007.

[8] D. Y. Huang, N. Apthorpe, G. Acar, F. Li, and N. Feamster, "Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale," *arXiv preprint arXiv:1909.09848*, 2019.

[9] 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.

[10] 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*, 2019, pp. 833–850.

[11] T. Liu, Z. Liu, J. Huang, R. Tan, and Z. Tan, "Detecting wireless spy cameras via stimulating and probing," in *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2018, pp. 243–255.

[12] K. Wu and B. Lagesse, "Do you see what i see?< subtitle> detecting hidden streaming cameras through similarity of simultaneous observation," in *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom*. IEEE, 2019, pp. 1–10.

[13] Y. Cheng, X. Ji, T. Lu, and W. Xu, "Dewicam: Detecting hidden wireless cameras via smartphones," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. ACM, 2018, pp. 1–13.

[14] "Best sellers in surveillance security cameras." [Online]. Available: https://www.amazon.com/Best-Sellers-Electronics-Surveillance-Security-Cameras/zgbs/electronics/898400

[15] "Best sellers in home automation devices." [Online]. Available: https://www.amazon.com/Best-Sellers-Home-Improvement-Automation-Devices/zgbs/hi/6478739011

[16] D. Ding, R. A. Cooper, P. F. Pasquina, and L. Fici-Pasquina, "Sensor technology for smart homes," *Maturitas*, vol. 69, no. 2, pp. 131–136, 2011.

[17] D. Sathyamoorthy, M. J. M. Jelas, and S. Shafii, "Wireless spy devices: A review of technologies and detection methods," *EDITORIAL BOARD*, p. 130, 2014.

[18] V. Valeros and S. 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* : 26*Nov*2019.

[19] C. W. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica: Journal of the Econometric Society*, pp. 424–438, 1969.

[20] H. Wu, K. Tan, J. Liu, and Y. 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*. ACM, 2009, pp. 67–76.

[21] X. Hu, L. Song, D. Van Bruggen, and A. Striegel, "Is there wifi yet?: How aggressive probe requests deteriorate energy and throughput," in *Proceedings of the 2015 Internet Measurement Conference*. ACM, 2015, pp. 317–323.

[22] S. Wenger, "H. 264/avc over ip," *IEEE transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 645–656, 2003.

[23] A. D. Singh, S. S. Sandha, L. Garcia, and M. 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*. ACM, 2019, pp. 51–56.

[24] R. Zhang and S. Cao, "Real-time human motion behavior detection via cnn using mmwave radar," *IEEE Sensors Letters*, vol. 3, no. 2, pp. 1–4, 2018.

[25] V. Kepuska and G. 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)*. IEEE, 2018, pp. 99–103.

[26] S. Kennedy, H. Li, C. Wang, H. Liu, B. Wang, and W. Sun, "I can hear your alexa: Voice command fingerprinting on smart home speakers," in *2019 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2019, pp. 232–240.

[27] C. V. Wright, L. Ballard, S. E. Coull, F. Monrose, and G. M. Masson, "Spot me if you can: Uncovering spoken phrases in encrypted voip conversations," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE, 2008, pp. 35–49.

[28] K. Gao, C. Corbett, and R. Beyah, "A passive approach to wireless device fingerprinting," in *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*. IEEE, 2010, pp. 383–392.

[29] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic classification through simple statistical fingerprinting," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 1, pp. 5–16, 2007.

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

[31] C. Zuo, H. Wen, Z. Lin, and Y. 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*. ACM, 2019, pp. 1469–1483.

[32] J. Ortiz, C. Crawford, and F. Le, "Devicemien: network device behavior modeling for identifying unknown iot devices," in *Proceedings of the International Conference on Internet of Things Design and Implementation*. ACM, 2019, pp. 106–117.

[33] 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*. ACM, 2017, pp. 506–509.

[34] 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.

[35] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens, "Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, 2016, pp. 413–424.

[36] N. Patel, "Dead reckoning, a location tracking app for android smartphones." [Online]. Available: https://github.com/nisargnp/DeadReckoning

[37] R. W. Levi and T. Judd, "Dead reckoning navigational system using accelerometer to measure foot impacts," Dec. 10 1996, uS Patent 5,583,776.

[38] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, 2006, pp. 27–35.

[39] S.-T. Sun, A. Cuadros, and K. 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*. ACM, 2015, pp. 3–14.

[40] L. Ojeda and J. Borenstein, "Personal dead-reckoning system for gps-denied environments," in *2007 IEEE International Workshop on Safety, Security and Rescue Robotics*. IEEE, 2007, pp. 1–6.

[41] P. Kinney *et al.*, "Zigbee technology: Wireless control that simply works," in *Communications design conference*, vol. 2, 2003, pp. 1–7.

[42] M. B. Yassein, W. Mardini, and A. Khalil, "Smart homes automation using z-wave protocol," in *2016 International Conference on Engineering & MIS (ICEMIS)*. IEEE, 2016, pp. 1–6.

[43] N. J. Muller, *Bluetooth demystified*. McGraw-Hill New York, 2001, vol. 1.

[44] J. C. Haartsen, "Bluetooth radio system," *Wiley Encyclopedia of Telecommunications*, 2003.

[45] Z. Li and T. He, "Webee: Physical-layer cross-technology communication via emulation," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, 2017, pp. 2–14.

[46] S. M. Kim and T. He, "Freebee: Cross-technology communication via free side-channel," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015, pp. 317–330.

[47] Nbc, "How to detect hidden cameras," Aug 2019. [Online]. Available: https://www.nbcmiami.com/news/local/How-to-Detect-Hidden-Cameras-in-Your-Vacation-Rental-530314101.html

[48] "Monitor wi-fi traffic - wireless bandwidth monitoring tools." [Online]. Available: https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/monitor-wifi-traffic

[49] P. Schmitt, F. Bronzino, R. Teixeira, T. Chattopadhyay, and N. Feamster, "Enhancing transparency: Internet video quality inference from network traffic," 2018.

[50] A. Steele, "Music revenue surges on streaming subscription growth," Sep 2019. [Online]. Available: https://www.wsj.com/articles/music-revenue-surges-on-streaming-subscription-growth-11567708974

[51] D. Kumar, K. Shen, B. Case, D. Garg, G. Alperovich, D. Kuznetsov, R. Gupta, and Z. Durumeric, "All things considered: an analysis of iot devices on home networks," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 1169–1185.

[52] Y. Sun, M. Liu, and M. Q.-H. Meng, "Wifi signal strength-based robot indoor localization," in *2014 IEEE International Conference on Information and Automation (ICIA)*. IEEE, 2014, pp. 250–256.

[53] X. Luo, W. J. O'Brien, and C. L. Julien, "Comparative evaluation of received signal-strength index (rssi) based indoor localization techniques for construction jobsites," *Advanced Engineering Informatics*, vol. 25, no. 2, pp. 355–363, 2011.

[54] W. Xue, W. Qiu, X. Hua, and K. Yu, "Improved wi-fi rssi measurement for indoor localization," *IEEE Sensors Journal*, vol. 17, no. 7, pp. 2224–2230, 2017.

[55] Z. Li, Z. Xiao, Y. Zhu, I. Pattarachanyakul, B. Y. Zhao, and H. Zheng, "Adversarial localization against wireless cameras," in *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications*. ACM, 2018, pp. 87–92.

[56] S. Jondhale, R. Deshpande, S. Walke, and A. 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)*. IEEE, 2016, pp. 594–598.

[57] D. G. Green, "Sinusoidal flicker characteristics of the color-sensitive mechanisms of the eye," *Vision research*, vol. 9, no. 5, pp. 591–601, 1969.

Appendix A Audio-based Localization for Personal Home Assistants

In this section, we describe the audio localization technique step-by-step. First, we place the source of the sound (smartphone playing a phrase containing the wake word of the device) at different points in the room and see how it affects the device traffic. Then we go around the room while SNOOPDOG repeats that sound continuously and checks them for

causality with device traffic as shown in Figure 11. Sound played at the points marked as green produces cause-effect relationship with the device traffic. We eliminate the regions 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.

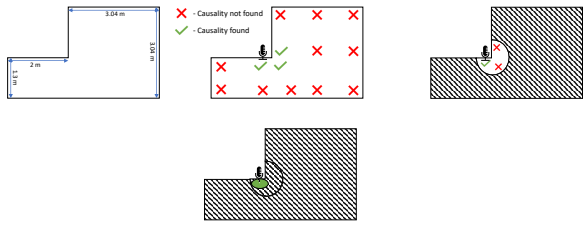


Figure 11: Trial-based localization for acoustic sensors.

Appendix B Traffic Variation of a Personal Home Assistant During Drop-In

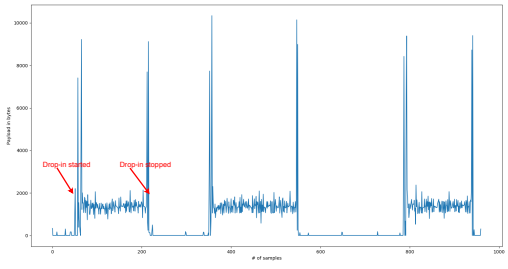


Figure 12: Traffic variation of Amazon Echo Dot while dropping-in

As discussed in the previous sections, Amazon Echo devices allow the user to drop into any of their Alexa devices and remotely listen to the audio in the room that they are placed in. This does not require any authentication on the device side during the drop-in. We perform 3 drop-ins on an Amazon Echo Device and show the traffic variation in Figure 12. From the traffic variation, it is clear when the drop-ins start and when they end.

Appendix C 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 [11, 57].

For camera encodings, we discard I-frames (through averaging), as they do not encode differences in a scene and require higher bandwidth, thereby adversely affecting the causality analysis.