UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

SMART SURVEILLANCE IN THE WILD: LOCALIZATION, INFERENCE, AND MANIPULATION OF WIRELESS CAMERAS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

BY

YAN HE
Norman, Oklahoma
2026

SMART SURVEILLANCE IN THE WILD: LOCALIZATION, INFERENCE, AND MANIPULATION OF WIRELESS CAMERAS

A DISSERTATION APPROVED FOR THE
SCHOOL OF COMPUTER SCIENCE

BY THE COMMITTEE CONSISTING OF

Dr. Song Fang, Chair

Dr. Li Song

Dr. Anindya Maiti

Dr. Chao Lan

Acknowledgements

First and foremost, I would like to thank my advisor, Dr. Song Fang, for his invaluable guidance, inspiration, patience, feedback, and continuous support throughout my Ph.D. journey. He is very nice and always there to help. I consider myself incredibly lucky to have him as my advisor.

I would also like to thank all my committee members, Dr. Song Li, Dr. Anindya Maiti, and Dr. Lan Chao, for dedicating their precious time to reviewing my dissertation and providing insightful comments and suggestions on my research. Additionally, I am grateful to the National Science Foundation (NSF) for its funding support.

To my labmates in the Cybersecurity research lab, thank you for all the interesting and inspiring discussions. They are Edwin Yang, Qiuye He, Guanchong Huang, Kepeng Zhou and Boyu Qu.

I would like to extend my appreciation to the University of Oklahoma and the School of Computer Science for providing me with the resources, facilities, and funding that have enabled me to conduct my research.

Last but not least, I owe a debt of gratitude to my family and friends. Their unwavering love, support, and understanding throughout my academic journey have been a constant source of motivation and inspiration.

Thank you all from the bottom of my heart.

This work incorporates material that has previously been published in peer-reviewed articles authored by me. Full citations of these publications are provided at the beginning of each respective chapter where the material appears.

Author Rights and Permissions

Portions of the research and findings presented in this dissertation have been previously published in peer-reviewed academic venues. Specifically, the material concerning *WeakCamID* was published in the ACM Conference on Computer and Communications Security (CCS) 2023 [87], and the work concerning *PhantomMotion* has been accepted to the Network and Distributed System Security Symposium (NDSS) 2026 [88].

In accordance with the respective publisher policies, the author retains the right to reuse these works. The Association for Computing Machinery (ACM) explicitly grants authors the right to reuse any portion of their published work, including the entirety of the accepted manuscript, in their own future works such as a thesis or dissertation. Similarly, the Internet Society (ISOC) and the NDSS Symposium operate under policies that permit authors to retain copyright and explicitly allow the inclusion of their published papers in a dissertation. The integration of these materials within this dissertation fully complies with the author rights and reuse policies of both ACM and NDSS.

Abstract

Wireless security surveillance systems are increasingly deployed in smart homes and other everyday environments due to their affordability, accessibility, and automated monitoring capabilities. Although these systems are designed to enhance safety, their reliance on wireless communication, cloud services, and embedded sensing components introduces new attack surfaces that threaten both security and privacy. This dissertation investigates two classes of non-intrusive attacks against modern wireless surveillance systems: passive service-state inference and active physical-world motion deception.

The first study shows that an adversary can infer whether a wireless security camera is operating without a paid cloud subscription by analyzing its external wireless traffic patterns. Such inference requires no compromise of the device, no access to the local network, and no interaction with the victim's cloud account, yet it reveals valuable information about the target's monitoring capability.

The second study demonstrates that motion-activated surveillance systems can be fooled into generating false motion events by injecting carefully controlled laser stimuli into their sensing regions and verifying system responses through wireless traffic sniffing. This attack does not require the adversary to physically enter the space monitored by the surveillance systems and remains effective across diverse commercial devices.

Together, these studies reveal that wireless surveillance systems can be exploited both as sources of information leakage and as targets of stealthy physical manipulation. By exposing these weaknesses, this dissertation advances the understanding of surveillance-system security and motivates the development of defenses against side-channel inference and sensor-level deception attacks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Over the past decade, the Internet of Things (IoT) has fundamentally reshaped pervasive computing. By seamlessly integrating sensing, computation, and wireless connectivity, IoT has brought unprecedented automation to both residential homes and commercial infrastructures. Among these connected devices, wireless security cameras occupy an especially critical role. Designed to provide remote surveillance, intrusion detection, and continuous property monitoring, these smart vision systems are trusted by millions of consumers to safeguard their most private spaces. However, this growing dependency introduces a profound irony: the very devices deployed to ensure safety can paradoxically become the weakest links, serving as vectors for privacy leakage and physical compromise.

The security posture of modern wireless surveillance systems hinges on two foundational layers: the *cyber layer*, which governs how the device communicates data to cloud servers over encrypted networks, and the *physical layer*, which

dictates how the device senses its immediate environment using hardware such as passive infrared (PIR) detectors and optical lenses. The overarching motivation for this dissertation stems from a critical observation: deeply embedded vulnerabilities within either of these layers can quietly, yet completely, subvert the system's security model. Often, these compromises occur entirely without the user's knowledge.

On the cyber front, while modern encryption standards (such as WPA2/WPA3 and TLS) are universally deployed to secure the payload of IoT communications, they fundamentally fail to obscure the underlying traffic metadata. Prior seminal research, such as the device type fingerprinting demonstrated by Miettinen et al. [126] and the foundational traffic analysis by Apthorpe et al. [28], has shown that packet timings and volumes inherently leak information about a device's identity and state. Simultaneously, on the physical front, while sensors are calibrated to specific operational thresholds (e.g., detecting human body heat), they are generally "blind" to the true source of an analog signal. As systematized by Giraldo et al. in their extensive survey on physics based attacks [68], rudimentary analog sensors lack the computational intelligence to authenticate whether an incoming physical stimulus is genuine or maliciously crafted. This dual-layered vulnerability severely challenges the perceived "safe state" of modern smart home environments, necessitating a comprehensive investigation into the novel attack surfaces introduced by these platforms.

1.2 Problem Statement

Despite advances in encrypted protocols and embedded hardware security, commercial wireless security cameras remain highly susceptible to structural flaws in

both network behaviors and physical sensor designs.

At the cyber layer, the industry shift towards subscription based cloud services has motivated manufacturers to implement differential behavioral models. Simply put, cameras operate fundamentally differently depending on whether a user pays for a premium cloud storage tier. This differential behavior inadvertently dictates the volume, frequency, and timing of encrypted Wi-Fi traffic. The open challenge lies in determining whether a passive, local eavesdropper can exploit these distinct traffic patterns to infer highly sensitive private states. If successful, an attacker could silently discern whether a camera is passively idling, actively recording an event, or currently feeding a live-view stream directly to its owner.

At the physical layer, the vast majority of wireless security cameras rely heavily on PIR sensors. These sensors are essential for minimizing power consumption, allowing the camera to remain in a deep sleep mode and waking up only when motion is detected. The fundamental problem is that PIR sensors are rudimentary analog components designed merely to react to localized fluctuations in ambient infrared heat. Consequently, they are inherently susceptible to targeted optical and thermal injections. The core challenge here is examining whether directed energy specifically, consumer grade lasers can be maliciously leveraged to spoof human motion. If adversaries can project deceptive thermal signatures, they could potentially bypass physical obstacles, utilize non-line-of-sight (NLOS) reflections, and manipulate the camera's operational state at will, all without leaving a conventional digital forensic trace.

1.3 Core Contributions

This dissertation identifies, evaluates, and demonstrates novel attack vectors against wireless security cameras from both network level side-channel perspectives and physical layer sensory spoofing. The core contributions are structured across two primary research thrusts:

1.3.1 WeakCamID: Network-Layer Privacy Leakage

We introduce *WeakCamID*, a universal camera state inference technique that leverages passive encrypted traffic analysis. Our findings highlight a vulnerability in how cameras without subscription shape their traffic. Due to the broadcast nature of wireless communications, eavesdropping is a classic security threat [59,61]. Even with rigorous encryption protocols in place, an adversary positioned within wireless range can passively sniff WiFi metadata. By intentionally generating localized physical motion to stimulate the camera, the adversary can correlate the resultant encrypted traffic bursts to successfully infer the camera's cloud subscription status and live-view mode. The significance of this contribution lies in exposing how commercial monetization models (subscription tiers) inadvertently act as side channels, compromising user privacy and revealing the exact operational capabilities of a target's defense system.

1.3.2 PhantomMotion: Physical-Layer Sensor Spoofing

Taking the investigation to the physical layer, we present *PhantomMotion*, a methodology illustrating how intentional laser injections can spoof PIR motion sensors. We demonstrate that adversaries can craft deceptive thermal and opti-

cal signals using visible or infrared lasers to trick the camera into "hallucinating" human motion. Crucially, *PhantomMotion* proves that such attacks do not require strict line-of-sight (LOS); by strategically heating opaque objects within the camera's field of view, an attacker can manipulate the camera's alert mechanism via non-line-of-sight (NLOS) thermal diffusion. This contribution reveals that smart surveillance systems fail to authenticate their environmental stimuli, exposing them to remote manipulation, reconnaissance, and false alarms.

1.4 Organization of the Dissertation

The remainder of this dissertation is organized as follows:

- **Chapter 2** surveys existing literature spanning IoT traffic analysis, encrypted privacy leakage, and physical sensor spoofing, drawing crucial distinctions between prior studies and our work.
- **Chapter 3** thoroughly details *WeakCamID*, describing the methodology for inferring camera subscription and live-view statuses through passive traffic fingerprinting.
- **Chapter 4** presents *PhantomMotion*, outlining the mechanics of laser-based motion injection attacks, their physical feasibility, and extensive real-world evaluations.
- **Chapter 5** discusses promising future research directions originating from our discoveries, particularly focusing on traffic obfuscation, sensor fusion defenses, and complex attack chains in smart homes.
- **Chapter 6** summarizes the key findings and concludes the research.

Chapter 2

Related Work

2.1 Privacy Leakage via Encrypted IoT Traffic

The proliferation of Internet-connected devices in home environments has attracted considerable attention from the security research community, particularly regarding the privacy risks posed by network traffic. While payload encryption is standard across modern Consumer IoT (CIoT) ecosystems, researchers have repeatedly demonstrated that encryption alone is insufficient to protect user privacy from passive observers.

Foundational work by Apthorpe [28] established that Internet service providers or local passive eavesdroppers can reliably infer the types of smart devices present in a home merely by analyzing send/receive traffic rates. Subsequent studies extended this paradigm to identify not only the devices but the specific user activities triggering them. For example, the *PingPong* [26] framework demonstrated how packet-level signatures which specifically the exchange of encrypted packets between devices and cloud servers that could pinpoint exact moments of

user interaction, such as a smart plug turning on or a door lock engaging. Similarly, *Peek-a-Boo* [14] proposed a machine learning-based approach to passively monitor and classify human behaviors taking place within smart homes by correlating encrypted traffic attributes (e.g., packet lengths and inter-arrival times) with specific physical activities.

While these studies effectively highlight the limitations of traditional encryption in shielding metadata, they primarily focus on generic activity inference or device fingerprinting. *WeakCamID* [87] distinguishes itself from this extensive body of literature by pivoting the focus onto system-level operational states tied directly to manufacturer subscription models. Instead of merely asking "is the camera active?", *WeakCamID* uses an active *stimulus-response* methodology. By injecting controlled physical motion into the environment and monitoring the resulting network traffic, our work infers whether the camera owner pays for cloud recording features and whether the owner is actively viewing the live stream. This approach exploits commercial API architectures and monetization strategies, presenting a novel side-channel threat not fully covered by classical IoT fingerprinting techniques.

Beyond camera state inference, prior research has extensively explored various dimensions of privacy leakage via wireless, visual, and sensory side-channels. For instance, wireless traffic analysis can be utilized to pinpoint hidden wireless cameras [86, 88], while vibration signals can be exploited to infer input PINs [90]. In the broader context of human activity and profile inference, wireless signals and other sensory leakages have proven to be surprisingly revealing. The survey [83] comprehensively reviews the privacy risks associated with inferring human profile information via wireless signals, highlighting the severity of such tracking. In particular, wireless channels can be utilized to identify location changes of a wireless

transmitter [58, 60] and can also carry sensitive information that enables covert communication [194]. Prior studies have further demonstrated that sensitive user input can be inferred without compromising target devices, including keystroke tracking over wireless channels [62, 192, 193], passcode inference in Virtual Reality environments using video-assisted approaches [96], and video identification through the analysis of blurred subtitle recordings [95].

2.2 Physical Sensor Spoofing in Cyber-Physical Systems

As smart environments become more reliant on physical sensors to drive autonomous decision-making, adversaries have increasingly targeted the analog-to-digital boundaries of these systems. Sensor spoofing involves injecting crafted physical phenomena such as light, sound, or electromagnetic waves to deceive a sensor into reporting fabricated measurements.

Initial research in physical sensor spoofing heavily targeted critical Cyber-Physical Systems (CPS) such as autonomous vehicles and drones. For instance, Yan et al. achieved contactless attacks against the ultrasonic and millimeter-wave (mmWave) radar sensors of self-driving cars, causing them to hallucinate obstacles or misjudge distances [190]. Optical sensors, particularly LiDAR arrays used in automotive navigation, have similarly been compromised. In *Illusion and Dazzle*, Shin et al. demonstrated how adversarial lasers could saturate or spoof LiDAR receivers, creating blind spots or phantom pedestrians [161]. Beyond autonomous vehicles, acoustic injection attacks have profoundly impacted embedded sensors. The *WALNUT* attack effectively disrupted the integrity of

Micro-Electro-Mechanical Systems (MEMS) accelerometers by broadcasting resonant acoustic frequencies, resulting in the destabilization of drones and the falsification of fitness tracker data [177].

2.3 Defensive Countermeasures and Mitigations

In response to both network and physical layer vulnerabilities, security researchers continue to propose theoretical and applied defenses.

To mitigate traffic analysis attacks like those demonstrated in *WeakCamID*, researchers advocate for traffic shaping and obfuscation. Strategies proposed by Apthorpe et al., such as independent link padding, aim to normalize the traffic flow exiting the smart home gateway, masking the bursty signatures associated with distinct events [27]. However, implementing these strategies directly on resource-constrained, battery-operated surveillance cameras remains challenging, as continuous dummy-traffic transmission inherently depletes battery reserves. In addressing advanced eavesdropping and inference threats across wireless networks, our team has pioneered proactive defensive mechanisms. This includes the strategic deployment of traps to detect and deter eavesdroppers [82], and ambush tactics utilizing defensive deception against wireless human monitoring, specifically neutralizing breath and crowd inference attacks [84, 85].

On the physical front, neutralizing analogue injection attacks relies heavily on multi-sensor fusion and algorithmic anomaly detection. Software-based approaches, such as *Sensor Defense In-Software (SDI)*, train machine-learning models to flag contradictions between redundant sensors or physics models [175].

However, many smart cameras deploy single-modality wakeup sensors (chiefly PIR) to conserve costs and power. As *PhantomMotion* exploits this very reliance on isolated sensor triggers, the implementation of complex sensor-fusion algorithms highlights an ongoing technological clash between manufacturing economics and proactive security design within Consumer IoT.

Prior research has also explored active deception and manipulation in sensing systems. Wireless sensing systems can be misled by carefully crafted signal-level effects [56, 63, 81], while wireless-based breathing and crowd inference can be disrupted through wireless deception [85].

This dissertation builds on and differentiates itself from these prior efforts by focusing specifically on modern wireless security surveillance systems, which combine wireless communication, cloud-backed services, motion sensing, and automated security decisions. Existing work shows that wireless and sensory side channels can reveal locations, keystrokes, profiles, videos, and user actions. In contrast, this dissertation studies how these risks appear in the specific and highly practical setting of home surveillance cameras. The first study shows that an attacker can infer whether a camera lacks a paid cloud subscription by observing wireless traffic, revealing whether important events may fail to be preserved [87]. The second study shows that an attacker can actively manipulate a motion-activated camera into reporting false events through laser-based motion injection, without disabling the camera or physically entering the monitored space [89]. Despite this deep exploration into automotive and flight control sensors, physical spoofing against residential surveillance systems that specifically Passive Infrared (PIR) sensors has remained largely underexplored. PIR sensors serve as the first line of defense for almost all battery-powered wireless cameras, guarding entry to higher-power visual recording states. *PhantomMotion* [89] bridges this

gap by demonstrating that inexpensive, consumer-grade lasers can successfully inject false motion signatures into smart cameras. Furthermore, unlike prior optical spoofing that predominantly necessitates direct line-of-sight (LOS), *PhantomMotion* advances the state-of-the-art by proving that thermal diffusion can be strategically leveraged against opaque materials to achieve non-line-of-sight (NLOS) triggers, silently bypassing structural boundaries. Together, these two studies bridge passive side-channel inference and active physical-world deception, showing that surveillance systems can both leak sensitive information and be manipulated against their intended purpose.

Chapter 3

Traffic-based Cloud Subscription Status Inference for Wireless Security Cameras

This chapter [1] introduces the existing vulnerabilities of wireless security cameras without subscription plans. Aiming at these vulnerabilities, it presents *Weak-CamID*, a novel traffic analysis technique that infers camera subscription status by correlating motion stimuli with network traffic patterns.

3.1 Introduction

Wireless security cameras are increasingly affordable, easy to install, and multi-functional (e.g., instantly alerting the camera owner to the presence of intruders

[1] I am using ACM authorship copyright for reusing this work. An early version of this work was published in Yan He, Qiuye He, Yao Liu, and Song Fang. "Traffic-based Cloud Subscription Status Inference for Wireless Security Cameras," in ACM CCS 2023.

and enabling the owner to converse with visitors). They have become an essential tool in a property protection kit, as they can help with the intrusion detection and the recovery of stolen items via video footage [179]. According to FBI data, an estimated 5,986,400 property crime offenses were committed in 2024. In the same year, law enforcement agencies made an estimated 910,654 arrests for property crime offenses. These figures indicate that a property crime occurred approximately once every 5.3 seconds [180].

Beyond the initial investment to buy the hardware, most wireless camera manufacturers offer consumers a paid plan to obtain more services, and offer limited functions for free users, so that users are motivated to pay for more services [104]. Usually, wireless cameras are equipped with motion sensors or microphones for enhanced protection, so that once motion or sound is detected, the camera is activated. The following behavior after the activation, however, often depends on whether the camera has an active subscription plan, which charges for services such as recording or cloud storage. For example, the latest Arlo cameras (e.g., Arlo Pro 3/4) do not actively record when events happen within their fields of view without a paid plan, and users can only get event alerts or manually stream footage to their smartphones via the Arlo app [70].

Cameras without paid subscriptions may suffer privacy issues, which have not been exploited before. We conduct a survey involving 220 participants: 213 of them believe the unpaid cameras can be used securely without privacy leakage; all users think the manufacturer guarantees that the system security is consistent across devices regardless of their subscription statuses. It is widely known that how owners safeguard their properties plays an important role when burglars select targets. A previous study reveals that in a panel made up of participants convicted of burglary, 13 out of 15 stated that they were not deterred by cameras

that they believed were not constantly monitored [54]. Similarly, if the knowledge is available, a burglar or other malicious user will likely first target properties whose cameras do not actively record and save videos.

Wireless cameras are usually in sleep/standby mode until motion is detected. A wireless camera (e.g., Arlo Pro 3) without a subscription often only sends a push notification about the event and then quickly returns to sleep mode. The network traffic correspondingly exhibits a short burst when an individual enters the motion detection range of the camera, and returns to normal after that. In contrast, when the camera has an active subscription, in addition to sending a push notification, the camera also records and uploads video to the cloud, which the owner can access later. The recording continues until motion ceases within the detection range, after which the camera reverts to sleep mode. Corresponding to this activity, there appears a long traffic burst lasting from the moment the person enters to when they leave the motion detection range. Both cases have distinguishably different wireless traffic patterns, which can be in turn utilized to infer the camera's subscription status.

In contrast to this immediate recording and upload, owners receiving push notifications via smartphones may or may not respond quickly or at all. As motion alerts are sometimes inaccurate or irrelevant, some users may disable notifications or become desensitized to them [78]. Generally, if they turn on the live view mode, the resultant live streaming will make the camera generate more traffic until the live view mode is turned off. Such a traffic burst may be confused with the one caused by the automatic cloud recording of a camera with a subscription. Nevertheless, a human cannot initiate the video processing module instantly when a push notification is received, as there are two non-negligible delays: (1) the user needs to first access the phone and tap the camera app,

depending on the user's response time; and (2) the app needs time to be launched. However, a subscribed camera can almost instantly begin cloud recording once it detects motion and sends the push notification. Consequently, the live mode and cloud recordings have different impacts on the traffic generation of the camera, and the resultant traffic pattern dissimilarity provides a clue to distinguish them.

We propose *WeakCamID*, a generalized framework to distinguish the state of a wireless camera, that is, whether or not the camera has a subscription and if its live view mode is turned on. Such an inference attack is non-trivial due to the following reasons. (1) The attacker cannot directly extract the traffic flow for the target camera, as it has neither control over the environment nor access to the WiFi network that the camera is connected to. The environment also likely contains many wireless devices and has a mixture of all flows from various devices, such as laptops, smartphones, or tablets. (2) To the best of our knowledge, previous extensive research efforts (e.g., [45, 86, 110, 114, 162, 189]) in detecting/localizing wireless cameras all assume that the cameras have the capability of recording the motion events without further distinction regarding the camera's subscription status. The existing course-grained traffic pattern identification methods can identify the existence of a camera while we cannot apply them to infer the subscription status. A novel and fine-grained traffic pattern technique is thus required. (3) Live streaming, relying on user operation, may generate comparable wireless traffic to cloud recording. Existing research either considers continuous/confirmed live streaming (e.g., [45,91,114,151,162]) or simply ignores it (e.g., [86, 112, 152]). To determine whether live streaming is on or off, human behavior then needs to be considered. (4) Traditional all-channel WiFi sniffing often requires rooted Android phones, limiting its practicality. Such an engineering challenge should also be overcome.

Almost all commodity wireless devices utilize 802.11 wireless protocols, and their use has an inherent weakness: exposure of link-layer Media Access Control (MAC) addresses [120]. A passive adversary within the radio range of a wireless camera can extract its MAC address, which tells the information of its device manufacturer via the beginning three most significant MAC bytes, i.e., the Organizationally Unique Identifier (OUI). *WeakCamID* first utilizes the motion-traffic correlation phenomenon to determine possible candidates of traffic flows belonging to a target camera, and then cross-references OUIs with publicly available manufacturer information to figure out the final candidate. By feeding motion stimuli to the camera and sniffing resultant traffic that varies with the camera state, *WeakCamID* builds a model to correlate motion-induced traffic with camera state. Such a model can be then used to map observed unlabelled traffic flows into corresponding camera states.

With *WeakCamID*, we discover that it could be counterproductive to install non-subscription wireless security cameras. The service differentiation between paying and non-paying users does not just create inequality in degrees of protection. The function restriction for non-paying users in fact introduces a serious vulnerability, which an adversary could take advantage of to identify properties with "weak" cameras. With a non-subscription camera, if the property owner does not view live streams in time, the events occurring in the area monitored by the camera will not be recorded. In such scenarios, as eye-witness descriptions and filmed recordings are not available, malicious users may perform inappropriate or criminal activities without worrying about being identified or leaving traces. We summarize our main contributions as follows:

- We point out the vulnerability of current wireless security cameras in differentiating services for paying and non-paying users, and develop the first

practical tool to successfully infer different camera states.

- We explore the correlation between motion stimuli with the resultant wireless traffic generated by cameras with varying subscription statuses.

- We conduct a comprehensive evaluation of both classical machine learning methods (Decision Trees, Random Forests, SVMs) and deep learning architectures (ANNs, CNNs, RNNs) for camera state inference. Our analysis reveals that SVMs achieve 95% accuracy with minimal inference time (8.6ms), while RNNs reach 99% accuracy at the cost of higher computational overhead (1228.5ms average).

- We show that *WeakCamID* can detect user response to motion alerts by distinguishing high traffic volumes caused by cloud recording and streaming.

- We develop an app for validating the effectiveness and efficiency of *WeakCamID*. Experimental results across 13 popular cameras (including two baby cameras) show that *WeakCamID* can attain a mean success rate of 95% to infer camera states within 19 seconds, with the RNN-based variant achieving up to 99% accuracy for enhanced precision when computational resources permit.

3.2 Background

Wireless Security Camera Subscription: Almost all wireless camera companies offer video-storage plans for customers to purchase. Compared to the traditional one-time revenue from hardware sales, recurring revenue from subscription plans is more predictable, sustainable, and potentially more profitable.

The subscription cost normally varies with the video resolution and the number of cameras supported. For example, Arlo offers two options for multiple cameras at a single home and on the same account, $9.99 and $14.99 per month enabling recording in up to 2K and 4K video resolution, respectively [33]. The seemingly small monthly charges, however, add up and may inevitably impose a financial burden on many users [149]. Non-subscription cameras often have limited features, such as live video streaming and motion notifications, rather than cloud recordings, which allow users to save captured videos on their online storage. In this paper, we point out that using such unpaid cameras is not as safe as paid versions and may cause significant privacy concerns.

Motion Detection: Wireless security cameras are often battery-powered, and most of them employ motion sensors to conserve battery, by only waking up when motion is detected. There are different types of motion sensors, including Passive Infrared (PIR), ultrasonic, microwave, tomographic, and combined types. Of these, PIR sensors are most prevalent, being small in size, cheap, and highly sensitive to motion. These are made of a pyroelectric film material sensitive to radiated heat power fluctuation [113]. This material generates electric signals when exposed to heat in the form of infrared radiation. Thus, PIR sensors can detect the presence of humans or other warm-blooded living beings from the radiation of their body heat [133], meaning that they can work even in the dark.

3.3 Attack Model and Assumptions

We consider a general scenario, where a wireless security camera is deployed to monitor a target area with an unknown subscription status. Once motion is detected in the camera's range, a camera without a subscription only sends a

push notification to the owner while a camera with an active subscription also enables cloud recording. After receiving push notifications about motion events, the owner may or may not turn on the live view mode of the camera through the camera app. The adversary aims to employ *WeakCamID* to infer the camera state, i.e., whether the camera has a subscription and if the live stream is opened.

We assume that the adversary has the capability to sniff wireless traffic and perform some probing motion in the target area. To avoid being exposed, the adversary can actively employ a helper or some moving robot (e.g., drone/robot/-car) that emits heat to introduce movement. Additionally, she can passively monitor camera activity and rely on others triggering motion sensing. Note that it is not necessary for the attacker to know the exact location of the camera. In a common scenario, people often make wireless security cameras visible with the hope of deterring malicious users. For example, owners may post signs and stickers to warn that there is a security camera present. Such visibility, however, could help the adversary quickly determine the possible motion detection range of the camera. On the contrary, when the camera is hidden, *WeakCamID* still works, as it can recognize the existence of wireless cameras by analyzing motion-induced wireless traffic. After the attacker confirms that the camera has no subscription and no live video is turned on, she may bypass it to perform further malicious activities (e.g., burglary and intrusion) without being recorded.

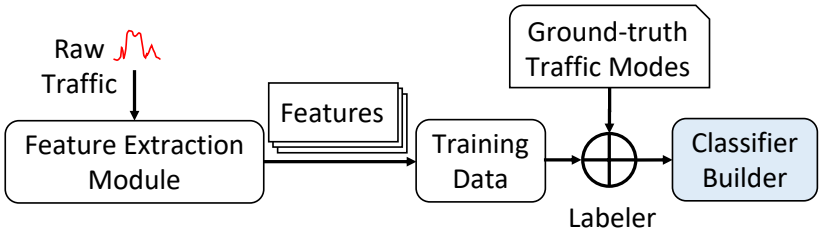


Figure 3.1: Offline training phase: building a traffic classifier.

3.4 Camera State Inference

3.4.1 Framework Overview

WeakCamID performs a two-phase process to infer camera states from observed wireless traffic: the *training* and *inference* phases. Figure 3.1 depicts the offline training phase, in which a traffic classifier is built with the motion-induced traffic data collected from sample wireless security cameras and their corresponding states. The inference phase then uses the trained traffic classifier to recognize new traffic flows, as shown in Figure 3.2.

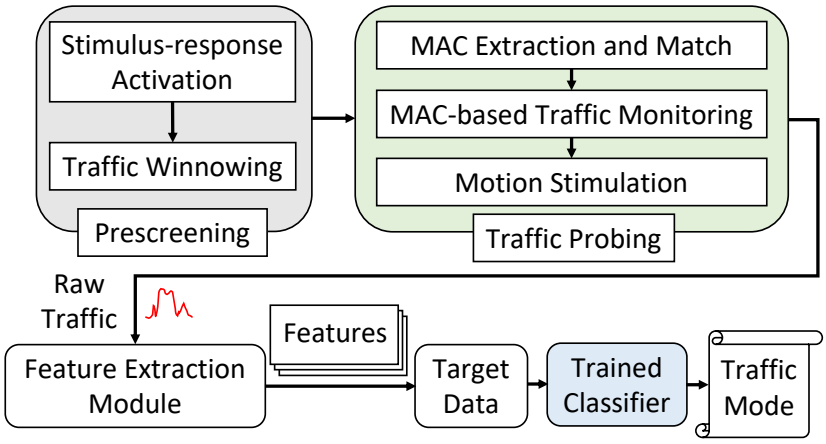


Figure 3.2: Inference phase: recognizing camera states.

As aforementioned, there may be a variety of devices sending out wireless

traffic in a new environment. Thus, in the inference phase, the adversary must first identify the traffic flow associated with the specific target camera. Toward the goals, *WeakCamID* introduces two important phases before extracting traffic features, which are *traffic prescreening* and *traffic probing*.

The first phase coarsely determines the wireless traffic flow associated with the target camera. When provoking motion within a target area, if there is a wireless camera monitoring this area, a corresponding wireless traffic burst will be immediately observed as the triggered camera generates traffic. The burst can be short or long depending on the camera's subscription status. The traffic flow exhibiting such a distinguishable pattern is regarded as a candidate.

In the second phase, we further eliminate the inference of other wireless devices which coincidently exhibit traffic patterns similar to the target camera in the first phase. We inspect the OUI in the MAC address embedded in each candidate traffic flow to sort out the camera-generated traffic flow and then monitor the surviving traffic. We then feed manipulated motion to the camera in order to extract features from the resultant traffic.

3.4.2 Training Phase

Our model is trained via data collection, feature extraction, state labeling, and traffic classifier building steps.

Data Collection: To capture raw wireless packets originating from the camera, we should know the channels that the camera operates on. The wireless network interface card (NIC) of a traffic sniffing device needs to be in monitor mode to listen to all the wireless traffic nearby. Generally, the monitor mode is disabled, and the default/normal mode of an NIC is managed mode, which makes

the device only capture packets with its own MAC address as the destination MAC and discard other packets.

Sniffing with Laptop: An intuitive way to achieve traffic sniffing is to use network sniffing software such as *WireShark* [66], but this method requires the sniffer to be able to access the same WiFi network as the target camera. The network is often secured with a password, which is unknown to the sniffer. Alternatively, if the laptop has a compatible wireless network adapter (e.g., Intel 622AN [98]) that supports monitor mode, the Aircrack toolkit [17], which is open source, can be then utilized to enable monitor mode.

Sniffing with Android Phone: A laptop may be bulky for a user to carry. To enable monitor mode on an Android phone, we need first to perform kernel live patching corresponding to the phone model [155] and then employ Airmon-ng tool [5], which is included in the Aircrack-ng package. For example, we enable the monitor mode on a Nexus 5 Android phone by using Nexmon [4] to patch the phone's kernel and run *WeakCamID* on the rooted Nexus 5. Finally, the collected traffic data are loaded into the SQLite database [13] for feature selection.

Feature Extraction: Different camera states usually lead to different spatiotemporal patterns in collected wireless traffic data. We then extract relevant features to construct our "feature vector", and use it to train the model.

Normally, when we continuously feed motion stimuli to a wireless camera for a period of time (e.g. 10 seconds), the camera experiences multiple phases. First, the camera sends an event notification to the owner, causing the first traffic burst. Second, the camera may or may not start recording the activity, depending on whether the camera has a subscription (i.e., cloud recording capability). If the camera has a subscription, it will immediately start to record the activity and upload the captured video to the cloud backend. As a result, another traffic

burst will be generated, which is often larger than the one appearing in the first phase. However, if the camera has no subscription, the camera will not record the activity, and the traffic volume will soon become zero after sending the event notification to the property owner. Finally, the traffic flow varies according to the action taken by the property owner after she or he receives the event notification, i.e., whether to enable the live video by opening the app associated with the camera. Specifically, if the owner quickly presses the push notification after receiving it and then opens the camera app to watch the live streaming video from the camera, the traffic throughput (i.e., the rate at which the wireless camera generates packets) will abruptly increase again. Accordingly, we refer to these three phases as *event notification*, *camera response*, and *user operation*, respectively. We then extract features unique to the camera state to characterize each phase, which collectively form a fingerprint that allows us to reliably identify the occurrence of a motion event from passively observed encrypted traffic.

Phase I - Event Notification: When motion is detected, a camera with a paid subscription (e.g., Arlo camera [29]) often generates a rich push notification, attaching a thumbnail image of the event to the event notification, while a non-subscription camera only sends the basic event notification. Thus, the corresponding peaks of the instant traffic throughput will differ. We define the period of this phase as from the beginning of the first observable traffic peak to the next one for paid cameras (as they start to record and upload the recorded video to the cloud), and back to 0 for unpaid cameras. We record two features: the peak traffic throughput T_1^p, and the mean throughput $\bar{T}_1$ over the phase duration.

Phase II - Camera Response: Paid cameras also perform cloud recording except for event notifications, while unpaid cameras do not and stay silent without being triggered. We regard the point from which the traffic abruptly increases or

decreases as the ending point of the second phase for paid cameras; the abrupt traffic change is determined by whether the live streaming is enabled or not. A user may not always respond to a notification, e.g., when they are busy or sleeping, while if the user chooses to turn on live video streaming, it needs some time, and this delay includes two parts: (1) the interval between the time when the user receives the event notification and the time when the user opens the app, and (2) the time that the app needs to load. Empirically, this delay is at least 3 seconds. For unpaid cameras, we just consider the period of the second phase as 3 seconds, and such a period is enough to characterize how unpaid cameras respond to motion after event notification. Similarly, we record the peak traffic throughput T_2^p and the mean traffic throughput value $\bar{T}_2$ in the second phase. Obviously, for unpaid cameras, we have $T_2^p = \bar{T}_2 \approx 0$.

Phase III - User Operation: This phase happens only when the user turns on live view mode. For paid cameras in normal mode (i.e., no live view is enabled), the generated traffic will be nearly stable until the motion ends, while in live view mode, such traffic becomes a combination of recording and streaming traffic and would thus be higher. Also, after the motion ends, there will be only streaming traffic until the user turns the live view mode off. For unpaid cameras, no recording happens in normal mode and thus no traffic is generated for it, while they are re-triggered to generate the streaming traffic in live view mode and revert to standby mode once the user closes the live view mode. We specify the third phase starting from when traffic burst appears after the second phase until when the camera enters standby mode. Likewise, we mark the corresponding peak traffic throughput T_3^p and the mean value $\bar{T}_3$ for this phase. If no live view is enabled, we set $T_3^p = \bar{T}_3 \approx 0$.

A camera's state has four possibilities, live view mode and normal mode (i.e.,

when live view is unopened), with and without a subscription accordingly. We refer to the four states as *Paid - Live View*, *Paid - Normal*, *Unpaid - Live View*, and *Unpaid - Normal*. The final feature-vector corresponding to the resultant wireless traffic when introducing motion stimuli to a wireless camera with one state (S_i, $i \in \{1, 2, 3, 4\}$) thus can be denoted with a 6-element vector, i.e., $FV(S_i) = [T_1^p, \bar{T}_1, T_2^p, \bar{T}_2, T_3^p, \bar{T}_3]$.

Impact of User Behavior: We do not assume deterministic user behaviors. The owner can make decisions arbitrarily. The success rate thus does not depend on user behavior, and *WeakCamID* works regardless of whether users respond to notifications. If the live view is off, the inference result would be 'paid-normal' or 'unpaid-normal'; otherwise, it is 'paid-live view' or 'unpaid-live view'.

State Labeling: Once the feature vectors are extracted from the sniffed wireless traffic, *WeakCamID* creates a training set by labeling camera states. The labeled feature vectors can be used to train the classifier in the next step.

Dataset Splitting: We have a dataset containing feature vectors coming from 13 different cameras that we examine for training. We perform different durations of motion from 2 to 16 seconds with increments of 2. For every motion length, we collect 70 corresponding traffic flows for each camera state of every camera, enabling us to obtain high inference accuracy. Thus, the built dataset has $13 \times 8 \times 70 \times 4 = 29,120$ feature vectors in total. We apply the common 80/20 split for training and test sets.

3.4.3 Traffic Classifier Building

The final component of our training pipeline is the construction of a supervised learning classifier to infer camera states from network traffic patterns. We adopt

machine learning over traditional statistical methods for two key reasons. First, different camera manufacturers employ proprietary protocols that generate distinct traffic patterns for identical events. For instance, Ring cameras transmit text-only push notifications, whereas Arlo cameras bundle thumbnail images with notifications, resulting in fundamentally different traffic signatures. Second, camera video streaming configuration difference between different cameras complicates universal modeling. For example, Arlo Pro cameras default to 1440p (2560×1440) while Arlo Ultra cameras use 2160p (3840×2160) [30], yet all Ring devices standardize on 1080p (1920×1080) [12]. Machine learning classifiers naturally accommodate such variations by automatically learning discriminative features from labeled data, eliminating the need for hand-crafted statistical models.

Classifier Selection. We evaluate two complementary families of machine learning algorithms to identify the optimal approach for camera state inference: *tree-based methods* and *neural network architectures*. Tree-based methods like Decision Trees (DTs) [144], Random Forests (RFs) [92], and Support Vector Machines (SVMs) [43] provide interpretable, efficient baselines with proven effectiveness on structured feature data. Neural networks like Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs), offer the capacity to learn complex non-linear relationships and temporal dependencies inherent in sequential traffic data. This dual strategy enables comprehensive comparison between classical and deep learning paradigms.

Tree-Based Classifiers. We implement tree-based models using scikit-learn [41]. Decision Trees (DTs) recursively partition the feature space based on camera's network traffic information, producing interpretable decision rules. Random Forests (RFs) extend this idea via ensemble learning, aggregating multiple decorrelated trees to improve robustness and reduce overfitting. For multi-class

classification across the four camera states, we adopt Support Vector Machines (SVMs) with a lightweight decomposition strategy. Specifically, we first train one binary SVM to distinguish paid vs. free cameras, and a second binary SVM to distinguish live-view vs. non-live modes. The final state is then determined by combining the two binary outputs, effectively covering the four classes with only two classifiers. This design reduces computational overhead compared to a full one-vs-one scheme while preserving accuracy. SVMs identify margin-maximizing hyperplanes in the feature space, providing strong generalization performance for our feature dimensionality.

Neural Network Classifiers. We implement neural network based classifiers in TensorFlow 2.10.0 [172] and tune them using a held-out validation set to ensure a fair comparison with tree-based and SVM models to reduce overfitting from network traffic data. For each architecture, we perform a small grid search over the number of hidden units, depth, learning rate, batch size, and dropout rate, in order to set the best training parameters. All models are trained with the Adam optimizer [102] using mini-batches which is good for time series data like network traffic data and we set the early stopping based on validation loss, and we use L_2 regularization to reduce overfitting.

The ANN consists of multiple fully connected layers with ReLU activations and dropout, which allows it to learn non-linear relationships between traffic features and camera states. The CNN uses one-dimensional convolutions and pooling along the temporal axis to capture local patterns such as structured changes in packet sizes and inter-arrival times, following the effectiveness of convolutional feature extractors in prior work [109]. The RNN adopts Long Short-Term Memory (LSTM) units [93] to model sequential dependencies across packet traces, making it well suited for mode inference when ordering and short-term dynamics

carry discriminative information. After tuning, these networks achieve competitive or high accuracy, but their longer inference time (on the order of seconds on constrained hardware) introduces non-trivial computational overhead, which we account for in our model selection.

Training Infrastructure and Hyperparameters. All models are trained and evaluated under identical experimental conditions to ensure fair comparison. Neural networks are trained on NVIDIA RTX 3080 GPUs (10 GB GPU memory) using the Adam optimizer [102] with learning rate $\eta = 0.001$, batch size 64, and maximum 100 epochs. We apply dropout regularization (rate 0.3) and early stopping (patience 10 epochs) to mitigate overfitting. The dataset is partitioned into training (80%), validation (10%), and test (10%) splits. Input features are standardized via z-score normalization. Tree-based models use default scikit-learn hyperparameters with 5-fold cross-validation for model selection. All experiments run on Python 3.8.10, with evaluation metrics computed via scikit-learn to maintain consistency across both algorithm families.

Classifier Selection: We evaluate all candidate classifiers on a held-out test set to identify the optimal model for camera state inference. Performance is measured through success rate (overall accuracy), precision, recall, and F1 score. For each camera state, we compute true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), yielding success rate $(TP + TN)/(TP + TN + FP + FN)$, precision $TP/(TP + FP)$, recall $TP/(TP + FN)$, and F1 score $2/(Recall^{-1} + Precision^{-1})$. We systematically analyze how motion duration impacts classifier performance, as the duration of the motion recording affects the amount of detectable features in training.

Tree-Based Model Performance.

Figure 3.3 presents success rates across varying motion durations for Decision

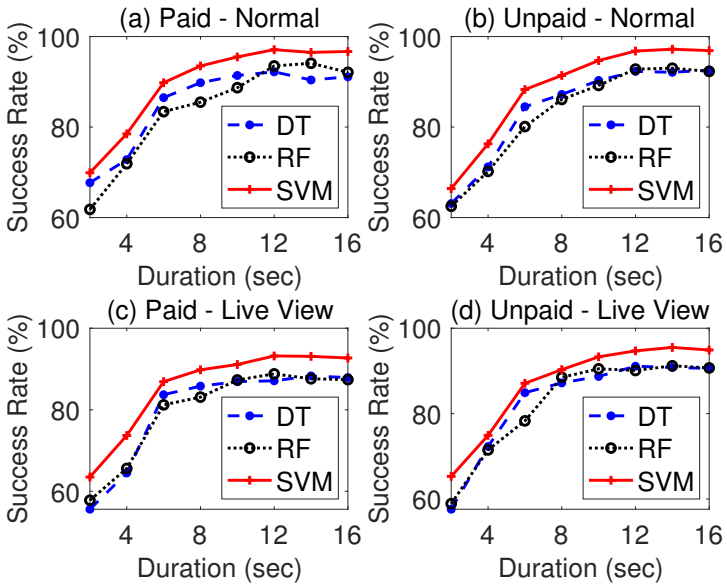


Figure 3.3: Comparison of success rates.

Trees, Random Forests, and SVMs. Three key patterns emerge. First, performance trends remain consistent across all four camera states, with normal-state cameras achieving marginally higher accuracy than those in live-view mode. Second, success rates improve monotonically as motion duration increases from 2 to 12 seconds, then stabilize. For durations below 8 seconds, all classifiers achieve less than 90% success rate, attributed to insufficient traffic volume for robust feature extraction. Beyond 12 seconds, success rates consistently exceed 90%. Third, SVMs demonstrate superior performance, achieving 97% success rate for both paying and non-paying cameras in normal state at 12-second motion durations.

Figure 3.4 corroborates these findings: SVMs attain an F1 score of 0.98 for normal-mode classification, indicating excellent precision-recall balance. Based on these results, we select SVMs as the strongest tree-based baseline for subsequent comparisons.

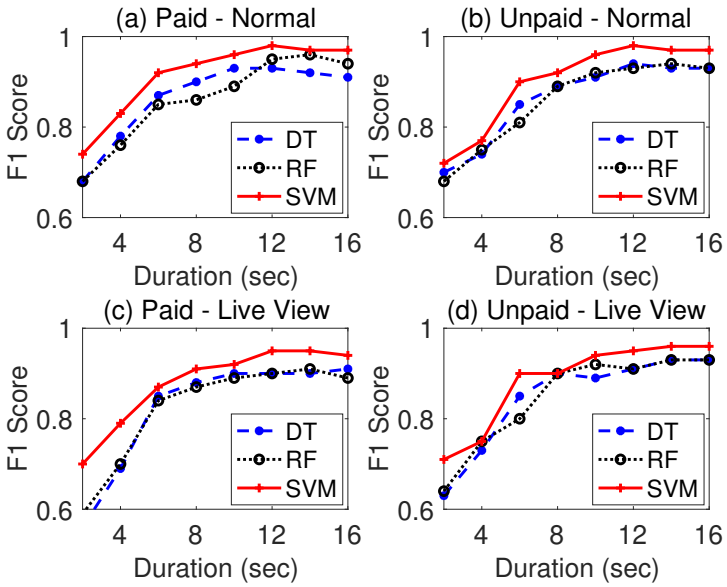


Figure 3.4: Comparison of F1 scores.

Neural Network Performance.

To assess whether deep learning architectures can further improve classification accuracy, we evaluate ANNs, CNNs, and RNNs on the same test set with identical preprocessing pipelines.

Figures 3.5 and 3.6 present success rates and F1 scores for neural network models alongside the best-performing tree-based method (SVM).

ANNs and CNNs underperform SVMs across all camera states. While ANNs can approximate complex non-linear decision boundaries through multilayer compositions, their feedforward architecture lacks mechanisms to exploit temporal ordering in sequential traffic data. CNNs, originally designed to extract spatial feature hierarchies from image data, prove ill-suited for our problem: camera traffic features exhibit strong temporal dependencies rather than spatial correlations, rendering convolutional filters less effective. The relatively modest performance gains of CNNs over ANNs (93% vs. 91% success rate) suggest that our features

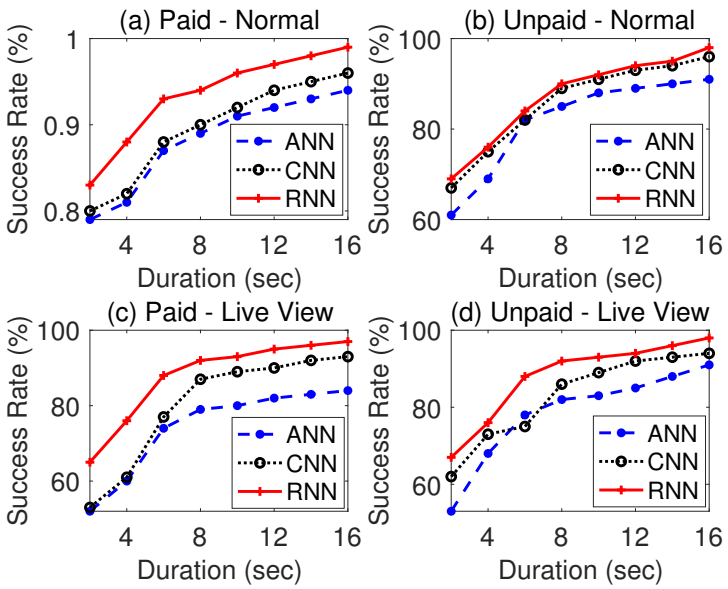


Figure 3.5: Comparison of success rates for neural network-based methods.

contain limited spatial structure amenable to convolutional processing.

In contrast, RNNs achieve superior performance, surpassing all other models including SVMs. At 16-second motion duration, RNNs attain 99% success rate for normal-state classification and maintain F1 scores above 97% across all four camera states. This substantial improvement stems from RNNs' inherent capacity to model temporal dependencies through recurrent connections and LSTM memory cells. Unlike SVMs, which construct static decision boundaries treating each observation independently, RNNs naturally capture how traffic patterns evolve during camera state transitions. The sequential structure of network traffic, where packet arrival times, sizes, and ordering encode state information which directly aligns with RNNs' architectural inductive bias. This finding corroborates prior work demonstrating RNN effectiveness for traffic classification tasks [145].

Efficiency Analysis. We also measures how long it would take for the machine learning model to infer the camera status after the specific model is

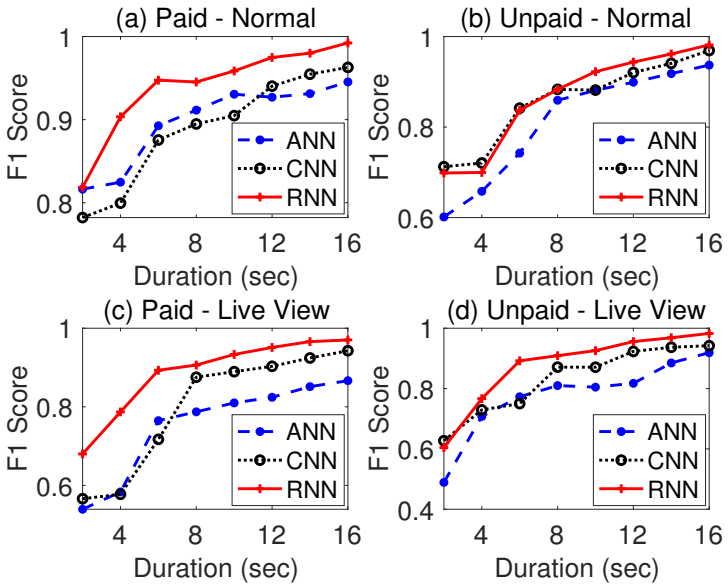


Figure 3.6: Comparison of F1 scores for neural network-based methods.

trained. Table 3.1 reports the mode inference latency for all evaluated models. For tree based models, Decision Tree and Random Forest provide fast predictions, with average latencies of 3.2 ms and 7.6 ms, but their detection accuracy is lower than the SVM. Neural network based models (ANN, CNN, and RNN) achieve strong accuracy, and the RNN reaches about 99% detection accuracy. However, they introduce a significant runtime overhead, with average inference times above 1,100ms per decision, which is costly for resource-constrained platforms such as smartphones or embedded cameras. In contrast, SVM offers a good balance, with an average inference time of only 8.6 ms while maintaining accuracy close to the RNN model. Considering both efficiency and accuracy, we select the SVM as the default classifier for all subsequent experiments.

Table 3.1: Mode inference time comparison across models (16-second motion window).

Model	**Min (ms)**	**Max (ms)**	**Avg (ms)**
Decision Tree	2.5	4.5	3.2
Random Forest	5.2	8.8	7.6
SVM	5.5	9.8	8.6
ANN	895.6	1425.2	1112.4
CNN	1258.5	1655.2	1415.3
RNN (LSTM)	1110.2	1420.9	1228.5

3.4.4 Inference Phase

In the inference phase, the adversary needs to first determine that the target camera is a wireless motion-activated camera via two important steps, traffic prescreening and traffic probing. The following processes are performed much in the same way as the training phase has, by attempting to achieve camera state inference through data collection, feature extraction, and traffic classification.

3.4.4.1 Traffic Prescreening

Over the air, there may exist diverse wireless traffic flows generated by a myriad of Internet-of-Things (IoT) devices or applications (such as smart TVs and digital voice assistants). We thus need to first distinguish the traffic flow of the target camera from traffic flows generated by non-camera devices and other wireless cameras deployed in the environment. We propose to generate motion (e.g., walking) within the camera's monitoring area to stimulate it, and then use the resultant wireless traffic to narrow down the candidates for the target traffic flow.

Stimulus-response Activation: Most wireless cameras are powered by rechargeable lithium-ion batteries, either built-in or removable. They normally sit in sleep/standby mode to save power consumption, and come awake when

(1) motion is detected or (2) the camera is manually turned on to live view. In standby mode, the camera usually just generates a "heartbeat signal" with a small size periodically (i.e., in order of seconds) to notify normal operation of the camera and synchronize with the base station or router.

Upon activation, the camera then sends a push notification of the motion event. If the camera has an active subscription, it also starts to record until motion stops and immediately uploads the video to the cloud for secure storage in the owner's library so that the owner can access them anytime; otherwise, if the camera has no subscription, only a push notification will be sent while no recording is initiated. Accordingly, abnormally high wireless traffic (indicating the push notification) will be generated regardless of the subscription status, and the traffic volume will soon become higher (as recording/uploading starts) for cameras with active subscriptions while decreasing to none (when heartbeat signals are ignored) for cameras with no subscription.

Therefore, to observe wireless traffic generated by the target camera, an adversary can feed the camera with activation signals by performing motion in the motion detection range of the camera.

Figure 3.7 depicts the traffic flow generated by a wireless camera (Ring Stick Up Cam with an active subscription) when we walk inside the motion detection range of the camera ($8 \sim 18$ sec). We observe that when the camera is in sleep mode, it only sends out a heartbeat signal of a small size. When the motion event is detected, the newly generated traffic volume suddenly increases immediately for sending a push notification. Next, as the camera starts to record to the cloud, a larger traffic volume appears until the motion in the motion detection range of the camera disappears. Without motion stimulus, the camera comes back to sleep mode.

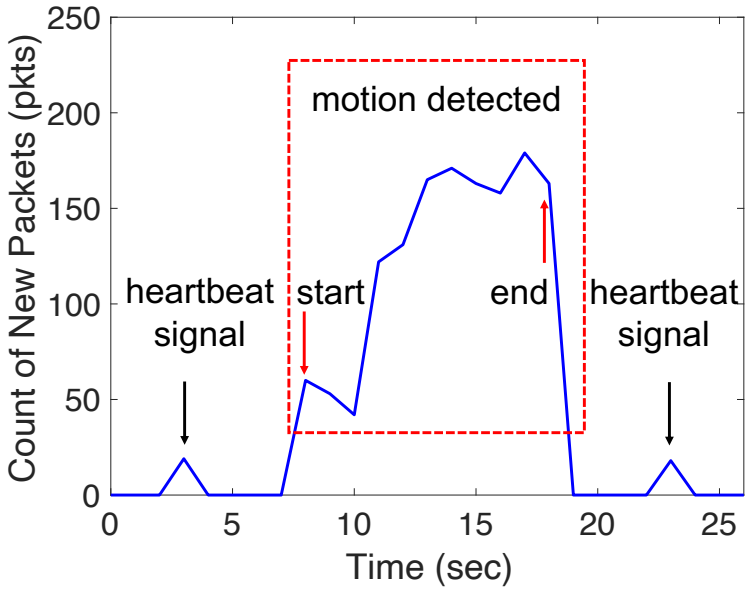


Figure 3.7: Traffic volume variation.

Traffic Winnowing: For a wireless camera in sleep mode (i.e., when there is no live view or video recording), the corresponding wireless microcontroller unit (MCU), such as TI (Texas Instruments) CC3220S [1] for a Ring Stick Up Cam, consumes low power and only listens for any trigger source. The motion sensor is integrated with the wireless MCU. Once it detects motion, it toggles the General Purpose Input/Output (GPIO) and generates an interrupt, which wakes up the camera to send a push notification and start cloud recording (if the camera has an active subscription). Consequently, the wireless traffic generated by a wireless camera has a strong correlation with the motion performed in the motion detection range of the camera regardless of the subscription status of the camera. Specifically, when a camera is wakened up by motion, a burst of wireless traffic can be immediately observed.

The distinguishable traffic pattern of the camera enables the adversary to winnow out irrelevant traffic flows, which do not show bursts according to the

appearance of the artificial motion. If a monitored wireless traffic flow suddenly jumps with the motion being performed and plummets as the motion stops, we then mark it as a candidate for the traffic flow of the target camera. As the environment may have multiple motion-activated devices including the target camera, one or multiple candidates may be identified.

3.4.4.2 Traffic Probing

It is essential to determine precisely which traffic flow belongs to the target camera before collecting its traffic features. We utilize the devices' MAC addresses to pinpoint the traffic flow associated with the target camera from the obtained traffic candidates in the previous step. After that, we set up a listener to monitor the traffic transmitted from the target camera and observe the traffic change on this channel when provoking the camera with manipulated environmental motion.

MAC Extraction and Match: A MAC address is a unique identifier assigned to a Network Interface Controller (NIC) for every networked device. It consists of 48 bits that are typically represented as 6 pairs of hexadecimal digits separated by colons or dashes. The first half is the Organizationally Unique Identifier (OUI), indicating a manufacturer or vendor; the second half refers to the device ID.

As IEEE 802.11 wireless communication (i.e., WiFi) employs security protocols such as WEP, WPA, WPA2, and WPA3 [153], the recorded videos are encrypted in WiFi signals. A general IEEE 802.11 MAC frame consists of a header, body, and frame check sequence (FCS). The header holds information about the frame; the body carries data that needs to be transmitted; FCS is used for detecting errors during the transmission. However, the header is unencrypted during transmission and exposes the MAC of the device sending the

traffic. For example, Ring Stick Up cameras utilize TI's chipset (i.e., CC3220S) for WiFi communication, and the OUI of their MACs starts with "40:BD:32", which indicates the SoC (System on Chip) from the manufacturer TI. The OUIs of different manufacturers are normally public [8]. We can thus build a dataset, referred to camera-tagged OUI database, containing OUIs of known vendors that manufacture wireless cameras.

WeakCamID first extracts the OUI in the MAC of each candidate for the traffic flow belonging to the target camera, and then checks the camera-tagged OUI dataset for a match of this OUI. If present, such a traffic flow is regarded as being generated by a wireless camera. Otherwise, it will be removed from the candidate list.

Dealing With MAC Spoofing: MAC addresses of NICs are hard coded in their circuit at the moment of manufacture. However, they can be changed via MAC randomization [65] or spoofing [100]. A camera may use a forged MAC with an OUI indicating a non-camera manufacturer for masquerading as a non-camera device, and similarly, a non-camera device may use a fake MAC with an OUI showing a camera manufacturer to pretend to be a camera. Since the payloads of raw WiFi packets are encrypted and the network of the target camera is inaccessible to the adversary, traditional traffic flow classification methods using a 5-tuple (source IP and port, destination IP and port, and IP protocol) or a 3-tuple (source IP, destination IP, and IP protocol) [121] do not apply. However, an attacker can launch the UUID-E (Universally Unique IDentifier-Enrollee) reversal attack [119, 181] to retrieve the original MACs for the devices with randomized or spoofed ones, as the UUID-E is derived from a device's original MAC and does not change with MAC. Alternatively, we utilize the wireless traffic pattern characteristics to uniquely identify camera devices.

The Systems on Chips (SoCs) are responsible for video/audio encoding and multimedia data transmission. Thus, the traffic patterns of a wireless camera highly depend on its SoC. However, the SoC choices are limited and most SoCs take largely identical operating flows, causing similar traffic patterns [45]. Particularly, wireless cameras follow universal standards to encode, encapsulate, and deliver video data to the cloud or users' devices. For example, Apple's HTTP Live Streaming (HLS), the most popular streaming format for the video industry according to an annual survey [39], requires that all videos must be encoded using H.264/AVC or HEVC/H.265 [24]. Accordingly, we train a Support Vector Machine (SVM) model by using the Scikit-learn libraries with Python 3.9, to distinguish traffic flows belonging to wireless cameras and non-camera devices.

An SVM classifier produces a hyperplane to best separate the input data into two classes. Since the cameras may or may not initiate video recording under different circumstances, the corresponding traffic patterns normally differ vastly. For such a multi-class case, we classify all traffic flows into three classes with the OVO (one-versus-one) approach [132]. For the cameras with no subscription and with live video mode turned off, they only generate traffic for push notifications and do not record video. We refer to such traffic as *Camera traffic 1*. For the cameras with subscriptions or with live video mode turned on, they also generate traffic for video recording, and we refer to the corresponding traffic as *Camera traffic 2*. We call the traffic generated by non-camera devices as *Other traffic*. We set a threshold according to the average data transmission rate of various wireless devices in the environment. For each traffic flow, we calculate its data transmission rate, as well as the difference between this rate and the threshold.

Figure 3.9 depicts the outcome of running the created multiclass SVM on a data set containing 800 traffic flows coming from wireless cameras (in different

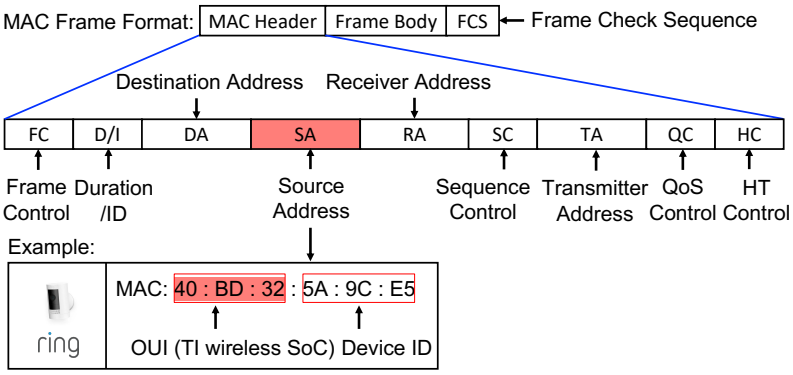


Figure 3.8: MAC frame format and a source address example.

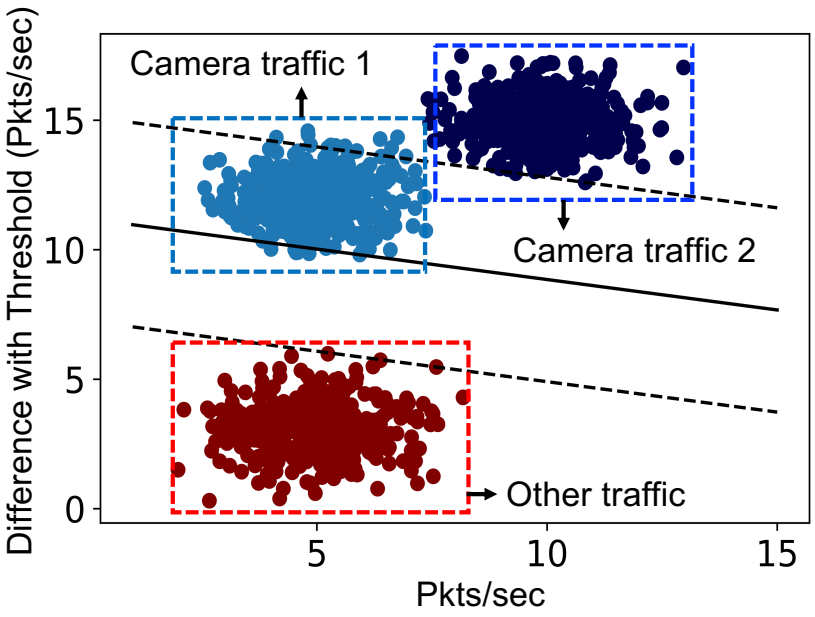


Figure 3.9: SVM classification results.

modes and subscription statuses) and non-camera devices, demonstrating the success of identifying traffic flows generated by wireless cameras.

MAC-specific Traffic Monitoring: By setting up a packet monitor with existing tools, we can listen to the traffic coming from the device identified as a camera. Particularly, we detect if the traffic volume varies and record the count change of intercepted packets.

Motion Stimulation: The longer we perform motion in the motion detection range of the camera with a subscription, the more (cumulative) packets the camera may generate. We have the same observation for the live view duration. We deploy four different wireless cameras (including Arlo Pro 3, Blink Outdoor, SimpliSafe Cam, and Wyze Cam Outdoor v2) to monitor the activity in an area. We perform two groups of experiments to verify the impact of cloud recording and live view on camera traffic, respectively.

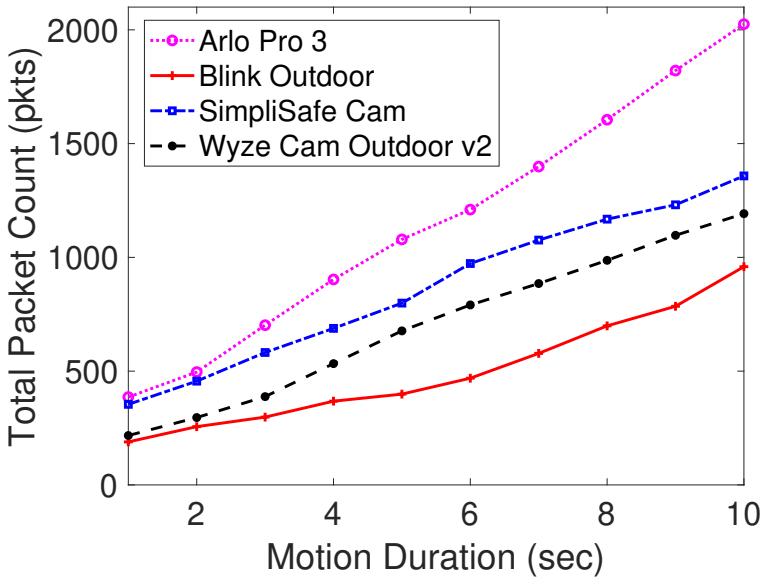


Figure 3.10: Impact of motion duration.

First, each camera has an active subscription and the live view mode is turned

off. We collect the traffic packets generated by each camera and count the corresponding total amount of the transmitted packets when a user manually introduces motion within varying durations, as shown in Figure 3.10. Second, each camera has no active subscription while the live view mode is turned on for streaming the activity. Similarly, we collect the traffic packets generated by each camera and count the corresponding total amount of the transmitted packets when the live view lasts different durations, as shown in Figure 3.11. We see that different cameras present diverse total packet lengths changing with the motion or live view duration, due to various recording or live streaming mechanisms taken by different camera manufacturers. Overall, the obtained total packet count (denoted with T) consistently shows a nearly linear correlation with the duration of both the motion and the live view. For example, for every second, the corresponding packet counts for Arlo Pro 3 to record to the cloud and to stream live videos are around 183 and 156, respectively. Accordingly, we consider a linear model to describe such as relationship, which is defined as follows,

$$T = c \cdot \Delta t + k, \tag{3.1}$$

where k is constant, Δt denotes either the motion or live view duration, and c represents the traffic throughput, i.e., the camera packets generation rate.

The model can be then utilized to determine whether the performed motion is still captured by the camera or whether the live view mode is still on. Specifically, if the observed total packet count and the motion or live view duration do not fit the linear model with a significant deviation, the cloud recording or live video streaming will be regarded as ended.

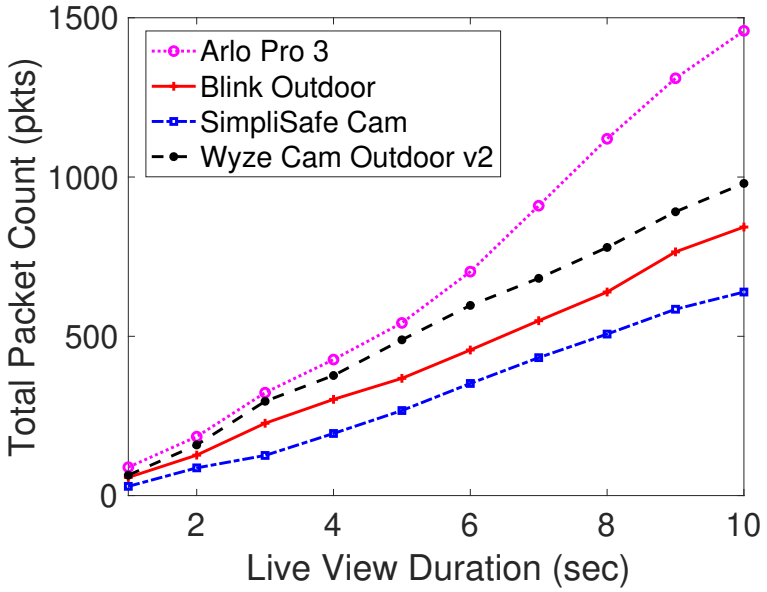


Figure 3.11: Impact of live view duration.

3.4.4.3 Traffic Inference

The process of traffic inference is defined much in the same way as the training phase by attempting to infer camera states via data collection, feature extraction, and traffic classification. After the traffic of the target camera responding to the motion stimuli is collected, the same features derived during training can be calculated. The obtained feature vector is then inputted into the built classifier, which outputs the camera state.

3.5 Implementation

3.5.1 Experimental Setup

We implement *WeakCamID* on commodity user devices. To achieve WiFi sniffing, existing studies often use rooted Android phones (e.g., [44,86]), while smartphone

Table 3.2: The list of wireless security cameras we test.

Camera ID	Model	Cloud Recording (Unpaid)
1	Arlo Pro 3	No
2	Arlo Pro 4	No
3	Arlo Ultra 2	No
4	Blink XT2	No
5	Blink Outdoor	No
6	Ring Stick Up Cam	No
7	Ring Spotlight	No
8	Reolink Argus 2	No
9	SimpliSafe Cam	No
10	Wyze Battery Cam Pro	No
11	Wyze Cam Outdoor v2	No
12	Hubble Nursery Pal Camera	No
13	Nanit Pro Baby Camera	No

vendors make it increasingly difficult to gain root access [168]. Instead, we design a new portable and low-cost external tool to enable WiFi sniffing, consisting of four components: a BLE module for a phone connection, a touch screen for user interaction, a WiFi adapter card (e.g., RTL8814AU chipset) in monitor mode, and a Raspberry Pi 4 Model B acting as a platform for the previous three components. This tool can connect with the developed mobile app *WeakCamID* via Bluetooth Low Energy (BLE). Our design makes it possible to run *WeakCamID* on any factory default smartphone without rooting it.

The app first scans the possible MACs for wireless cameras. The adversary then performs motion to stimulate the camera. The app logs accelerometer readings for motion speed calculation. With observed traffic, the app outputs the current camera state and the consumed time, indicating completion of status determination. We test 13 most popular wireless cameras (Including two baby cameras with ID 12 and 13), as shown in Table 3.2. Such camera models are selected from major brands sold online on Amazon and BestBuy. Non-paying

cameras only have basic functions (live video streaming and event notification) while paying ones offer cloud recording capability. We therefore consider two typical scenarios:

- *Outdoor:* The camera is mounted on the front outside wall (height: 10 ft; width: 17 ft) of a typical American single-family house to monitor the entryway into the house.
- *Indoor:* The camera is installed on the wall of a living room (of 372 sq. ft) to monitor the room.

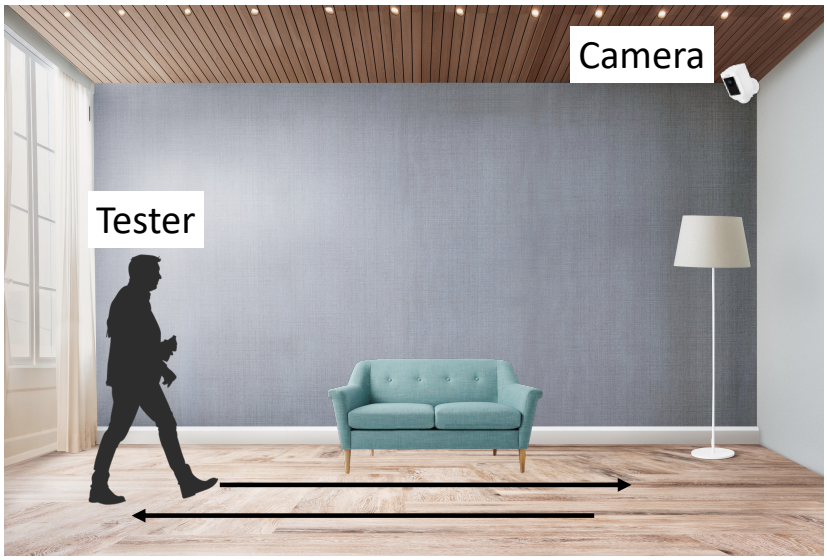


Figure 3.12: Layout of the experimental environment: Indoor.

Figures 3.12 and 3.13 show the layout of the two environments, where the camera is deployed with its field of view not blocked by the wall or other obstacles and an adversary can thus feed motion stimuli to it.

Evaluation Metrics: We use the following three metrics.

- *Suceess rate:* this is the ratio between the number of successful camera state inference attempts and the total number of inference trials.

Figure 3.13: Layout of the experimental environment: Outdoor.

- *F1 score:* this is the harmonic mean of precision and recall, with its best value at 1 and worst score at 0.
- *Detection time:* this is the amount of time spent on obtaining the camera state in terms of the subscription plan and live streaming mode.

3.5.2 Case Study

We use two Arlo Pro 3 cameras (one with and the other without a subscription) to monitor the same area, as shown in Figure 3.12. The user determines there exist wireless cameras monitoring the area, initiates motion in the area, and sniffs environmental wireless traffic. We test the following three situations.

No Live Video is Streamed on Both Cameras: When the user does not notice the motion notification (e.g., the phone is muted), no live stream will be opened. Figure 3.14 (a) shows the traffic flow generated by the two cameras. We observe a strong correlation between the traffic volume (i.e., count of newly

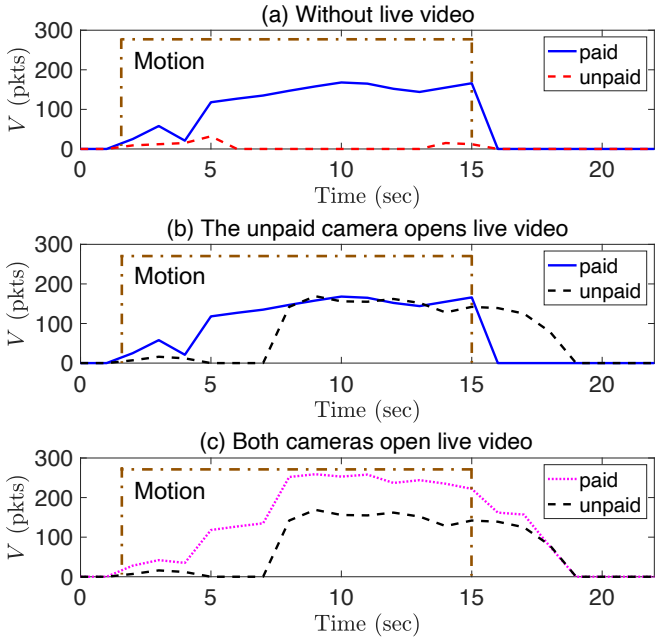


Figure 3.14: Traffic volume V (pkts) changes.

generated packets) with the motion for the paid camera, i.e., the volume matches with the newly performed motion. However, for the unpaid camera, there is only a small amount of traffic at the beginning of the motion, corresponding to the motion notification. The paid camera not only sends a notification but also records to the cloud until the motion ends. Furthermore, we see that the traffic volume for a motion notification of the paid camera is larger than that of the unpaid one. This is because, with a subscription, the push notification information is richer and includes a thumbnail image from the recorded video [29], which is not available for the unpaid camera.

Live Video is Streamed Only on the Unpaid Camera: Just for the unpaid camera, we stream live video once receiving the motion notification. Figure 3.14 (b) compares the corresponding two traffic traces and we see clear differences. First, unlike the paid camera, which automatically records after being activated by the motion, the unpaid one re-generates the traffic burst only after

the live view is turned on (at the 8^{th} second). To stream live video, we have to tap the notification or the app on the phone. Human reaction, tapping, and app login take time. There is thus an inevitable delay between detecting the motion and the start of the live video stream. Second, we may not end the live video exactly as the motion ends. We habitually watch until the motion ends and then close the app. Similarly, we need time to react and close the app. That's why we still observe traffic burst even after the motion ends for the unpaid camera. However, the paid camera ends recording (i.e., generating traffic bursts) precisely once the motion ends.

Live Video is Streamed on Both Cameras: Figure 3.14 (c) shows the traffic volume of both cameras streaming live videos. Unlike the unpaid camera, the paid one generates high traffic volume immediately once the motion is detected. Also, when the motion lasts and the live video is on, the traffic volume for the paid camera is apparently higher than that for the unpaid one. This is due to the fact that the paid camera streams live video and uploads the recorded video to the cloud at the same time, while the unpaid camera only streams live video. These results convincingly verify that the two cameras' traffic traces in this situation are still distinguishable.

By extracting features from the observed traffic flows, *WeakCamID* is able to successfully infer these camera states.

3.5.3 Impact of Motion Duration

Different durations of motion occurring within the motion detection area of the camera (with a subscription or in live view mode) may generate varying wireless traffic volumes. Accordingly, we vary the value of motion duration from 8 to 16

seconds, with increments of 2 seconds. For each value and camera state of every camera, we perform 10 trials and have $13 \times 4 \times 5 \times 10 = 2,600$ attempts in total.

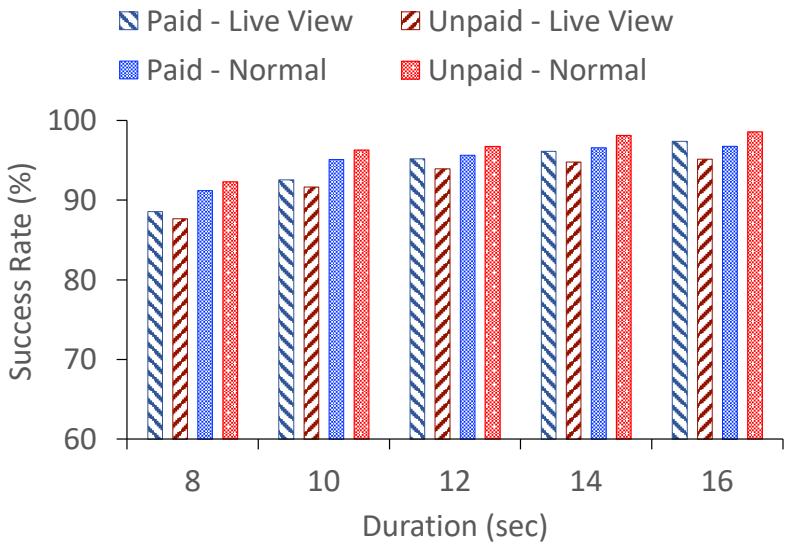


Figure 3.15: Effect of motion duration on detection success rate.

Figure 3.15 shows the average success rates for different motion durations. We have the following observations. First, the success rate always maintains at a high level, i.e., ranging from 88% to 99%, regardless of motion duration and camera state. Second, with the duration increasing from 8 to 12 seconds, the success rate becomes larger. It then maintains a stable high value (above 94%) after the duration is longer than 12 seconds. Lastly, the unpaid camera in live view mode and the unpaid camera in normal mode consistently has the lowest and highest average success rates regardless of motion duration.

This appears as the motion-induced traffic flows generated by unpaid cameras in live view and normal modes are the least and the most distinguishable, respectively. Figure 3.16 presents the F1 scores for all varying motion durations. We see that the F1 score is always above 0.9, again indicating high inference accuracy.

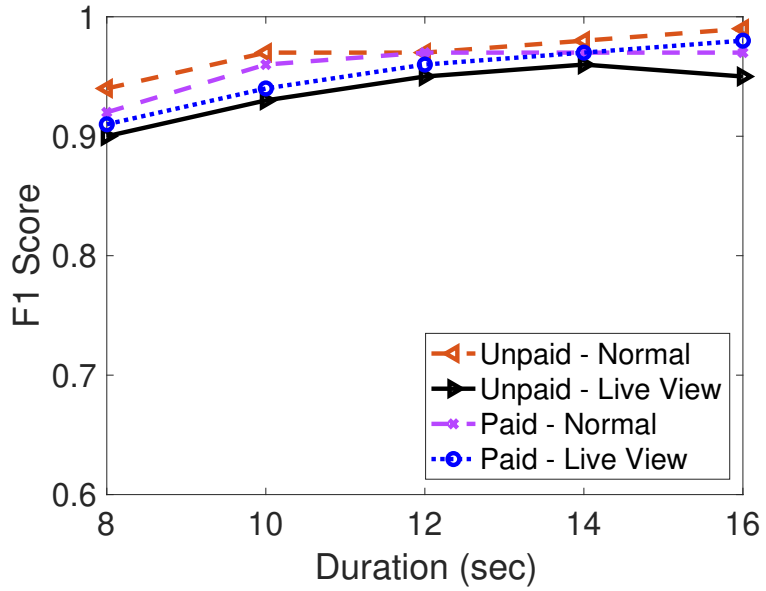


Figure 3.16: F1 score vs. motion duration.

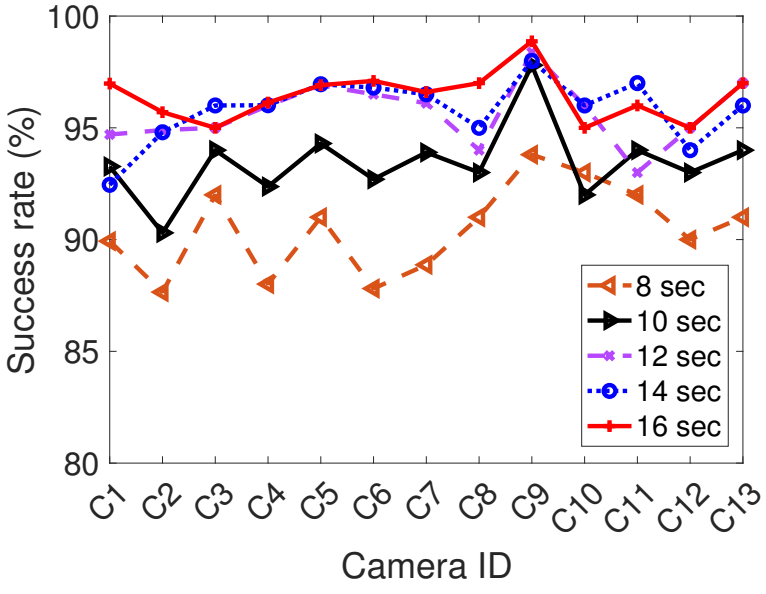


Figure 3.17: Average success rates across cameras.

Figures 3.17 and 3.18 present the average success rates and F1 scores of all camera states for each camera with varying motion durations. We see that the success rates and F1 scores for all cameras are consistently high (with a minimum of 88% and 0.89), while C9 (SimpliSafe Cam) always has a higher success rate or F1 score than the rest. This appears because C9 uses differentiated video streaming quality for paid and unpaid cameras while others use the same quality for both types of cameras.

The resolution of C9 is 1080p (1920×1080) with a subscription and decreases to just 480p (640×480) with no subscription. Such difference further enlarges the discrepancy between corresponding traffic volumes, facilitating camera state distinction. Also, we find for most cameras, the success rate or F1 score increases with the motion duration until the latter reaches 12 seconds, and remains relatively stable after that.

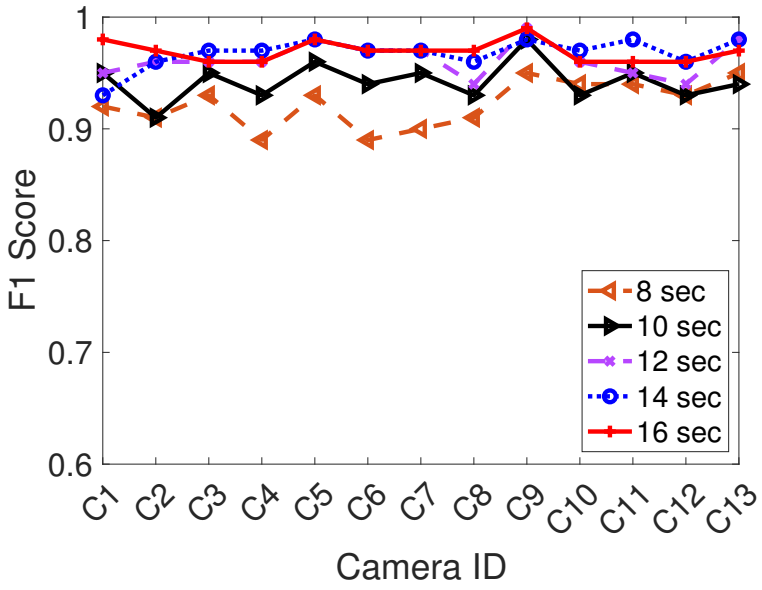


Figure 3.18: F1 scores across cameras.

3.5.4 Impact of Movement Speed

The speed v_m of motion occurring in the camera's detection range may affect its recording behavior. For example, if the speed is too slow, from the camera's perspective, the total motion may consist of multiple short activities. Compared with a quick motion which just triggers the camera once, such a slow one may cause the camera to be activated multiple times in a discontinuous way. We vary v_m from 0.2 to 1.4 meters per second (m/s), with increments of 0.2. The app logs accelerometer readings for calculating the speed. For each v_m and camera state, we perform 100 attempts of *WeakCamID* to infer the state of the camera (Ring Stick Up Cam).

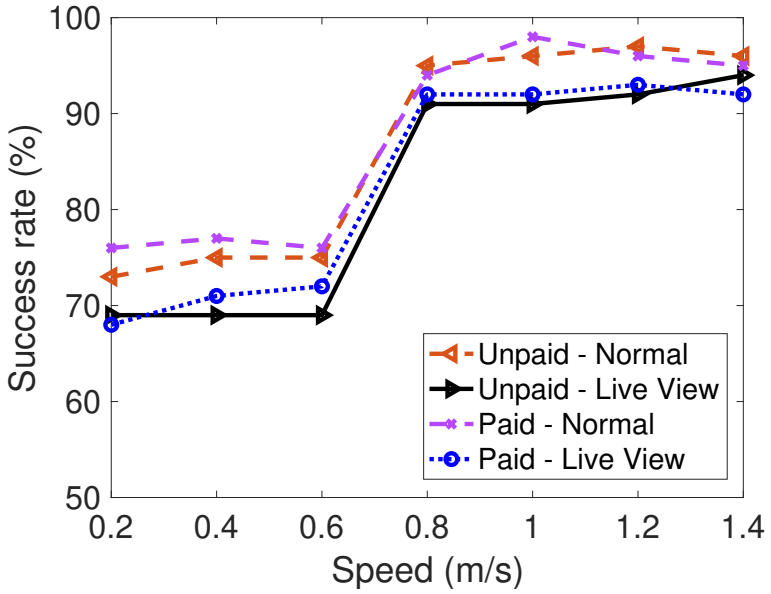


Figure 3.19: Impact of movement speed.

Figure 3.19 illustrates the average success rates when v_m varies. We observe that the success rate is below 77% when v_m is no larger than 0.6 m/s. This is because the low speed may trigger the cameras multiple times and cause the

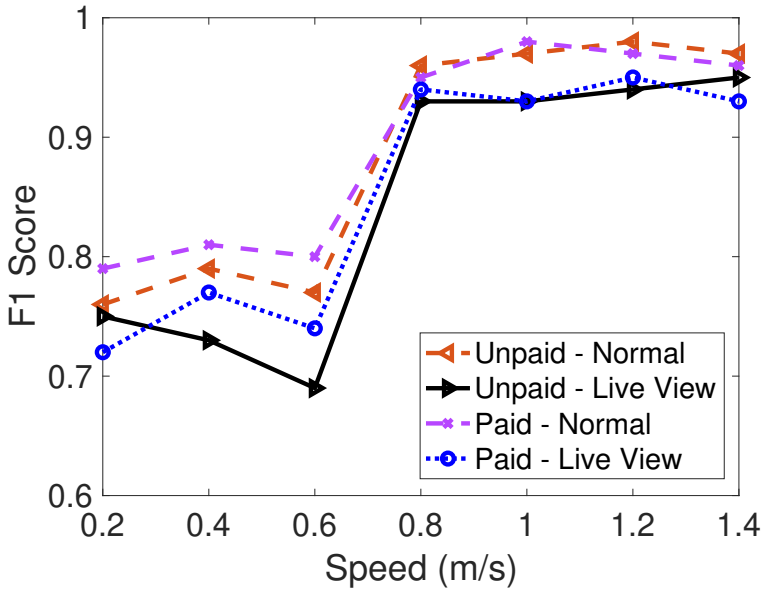


Figure 3.20: F1 score vs. speed.

camera to generate multiple notification alerts. The resultant traffic patterns become less discernible. Also, we see that once the walking speed reaches 0.8 m/s, the success rate can always be larger than 92%. Meanwhile, for the same speed, the corresponding success rates for normal mode are consistently higher than that for live view mode. Specifically, the average success rates for normal and live view modes are 96.0% and 92.5%. This appears due to the fact that the live view mode is controlled by the user, who may turn it on at a random time after receiving a motion alert, causing the traffic patterns associated with streaming live videos more diverse. Figure 3.20 plots the corresponding F1 scores, which always exceed 0.93 when the speed reaches 0.8 m/s. Also, the F1 scores for the normal mode are higher than that for the live view mode. The range for normal walking speed is 1.2 to 1.4 m/s for adults [67]. *WeakCamID* can thus achieve high accuracy without requiring an average user to change gait speed.

3.5.5 Impact from Resolution

One of the main reasons for the difference in traffic volume across cameras is the resolution of the uploaded video. For example, sending a 4K (3840×2160) video needs about four times as much bandwidth as sending a 1080p (1920×1080) video. In practice, a camera may switch among several resolutions when uploading video to the cloud, which can change the traffic volume over time. To study how video resolution affects the inference accuracy of our method, we ran extra experiments with an Arlo Ultra camera. This camera supports several resolutions: 4K (3840×2160), 2K (2560×1440), 1080p (1920×1080), 720p (1280×720), and 480p (640×480). Our goal was to see whether the resolution setting has a strong effect on the accuracy of WeakCamID.

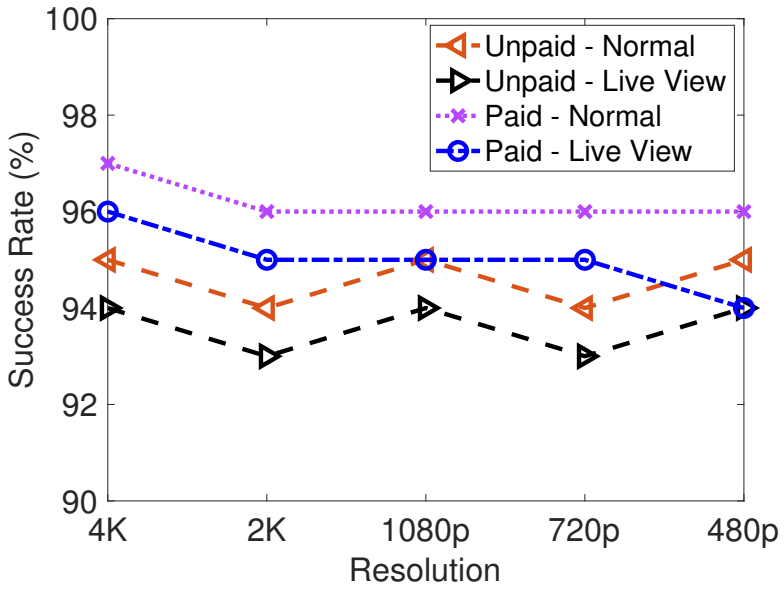


Figure 3.21: Success rates vs. resolutions.

Figure 3.21 shows the success rates for the five resolution settings. The inference accuracy stays high, with success rates above 94% at all resolutions for both

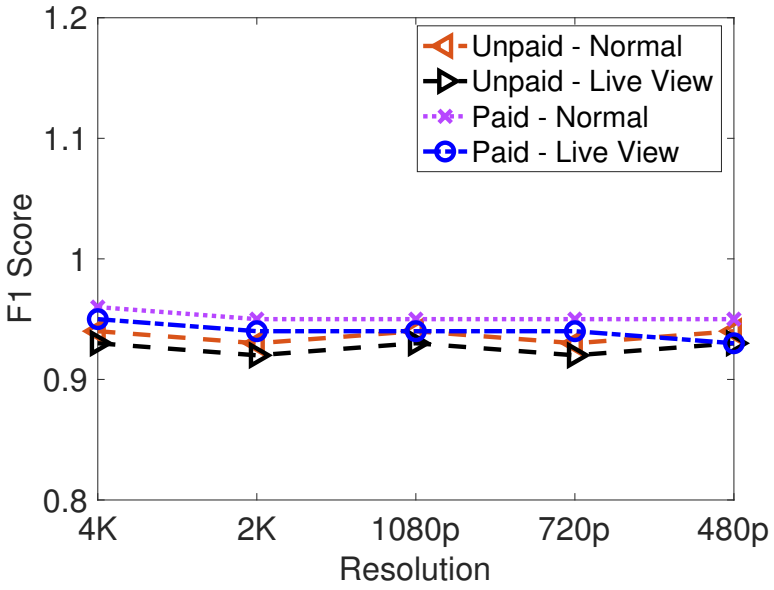


Figure 3.22: F1 scores vs. resolutions.

Paid normal and Unpaid normal modes, and above 93% for both Paid live view and Unpaid live view modes. Figure 3.22 shows the related F1 scores, which are above 0.93 for normal modes and above 0.92 for live view modes. These results suggest that changes in video resolution have a very small effect on the overall inference accuracy of WeakCamID. Even at lower resolutions such as 480p and 720p, the accuracy is close to that at higher resolutions. This is because changing the resolution does not greatly change the key differences in traffic volume between paid and unpaid camera states.

3.5.6 Impact of Previously Unknown Cameras

One concern is whether our model works for a new camera, whose brand/model is previously unknown. As aforementioned, most camera vendors take largely identical operating flows, causing their traffic variation quite consistent. *WeakCamID* can be thus applied to infer states of new cameras without retraining the

model.

We specify one camera as the new camera and use the other twelve (in Table 3.2) for training. Accordingly, we generate 13 traffic classifiers, referred to as *Victim-exclusive*. We then use each classifier to infer the state of the corresponding new camera 100 times, whose traffic data are not included in the training data set of this classifier. For comparison, we also investigate the performance of the classifier (called *Victim-inclusive*) that utilizes all 13 cameras for training, and use it to infer the state of every camera 100 times.

Figure 3.23: Accuracy comparison for new cameras (victim-exclusive vs. victim-inclusive).

Figure 3.23 presents the comparison of the success rates for applying *Victim-inclusive* and *Victim-exclusive* classifiers. We see that the *Victim-inclusive* classifier always performs slightly better than corresponding *Victim-exclusive* ones. Specifically, the mean success rate for all *Victim-exclusive* classifiers is 94.1% while the *Victim-inclusive* classifier achieves an average success rate of 95.8% across all cameras.

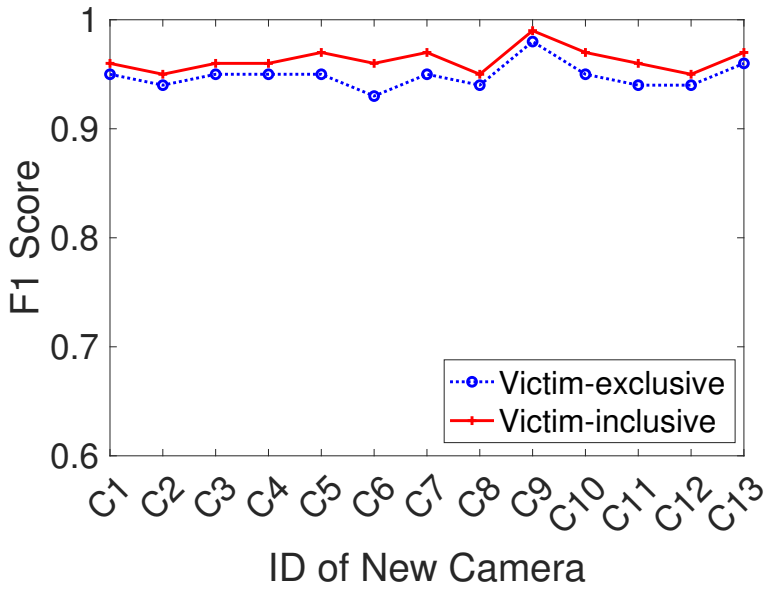


Figure 3.24: Detection time for new cameras (victim-exclusive vs. victim-inclusive).

Figure 3.24 compares the average detection time. We observe that for each victim camera, the detection time obtained from the *Victim-exclusive* classifier, ranging from 16.6 to 18.9 seconds, is always slightly longer than that obtained from the *Victim-inclusive* classifier. The small increase in detection time comes from requiring a longer time for the corresponding *Victim-exclusive* classifier to process the data. These results show that *WeakCamID* works for new cameras with a high probability and within a short period.

3.5.7 Overall Inference Performance

For each mode of every camera in Table 3.2, we perform 100 trials in each environment. Thus, we have $13 \times 4 \times 2 \times 100 = 10,400$ attempts in total.

Figures 3.25 and 3.26 present the success rates for different cameras in the indoor and outdoor environments. We observe two major tendencies. First, the

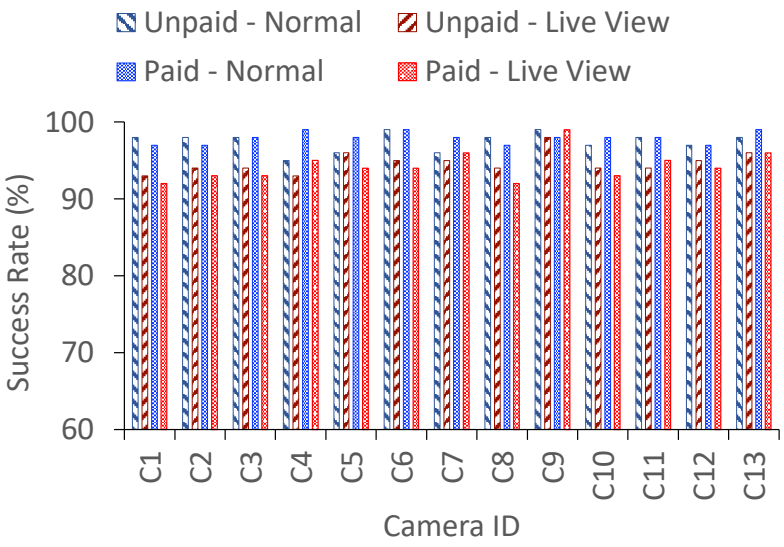


Figure 3.25: Overall inference success rates (indoor).

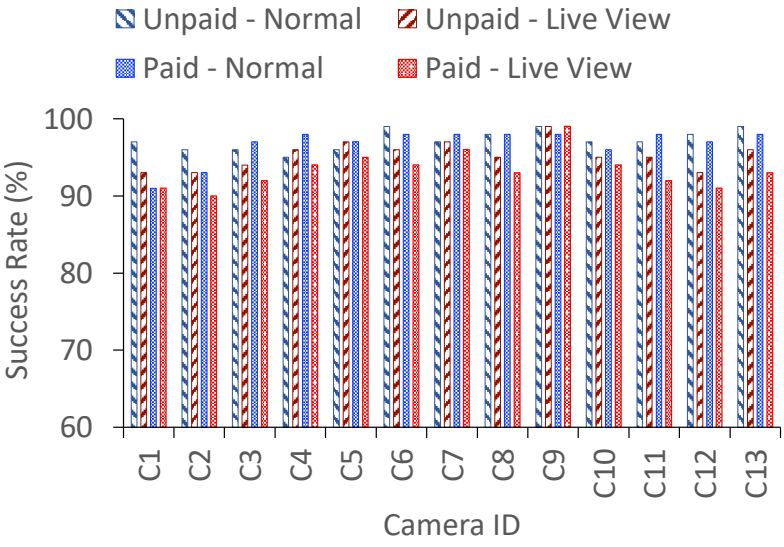


Figure 3.26: Overall inference success rates (outdoor).

success rate is consistently high over different camera states and models, ranging from 92% to 99% and 90% to 99% for the indoor and outdoor environments, respectively. Particularly, for C9 (SimliSafe Cam) in both environments, our technique can detect all camera states with a success rate always above 98%. This again confirms that the recording quality differentiation strategy taken by C9 makes traffic flows more distinguishable. Second, a camera in normal mode can usually be detected with higher accuracy, especially when the camera has a subscription. This may be because cameras in live view mode generate higher traffic volume, causing the traffic flows to be misclassified more.

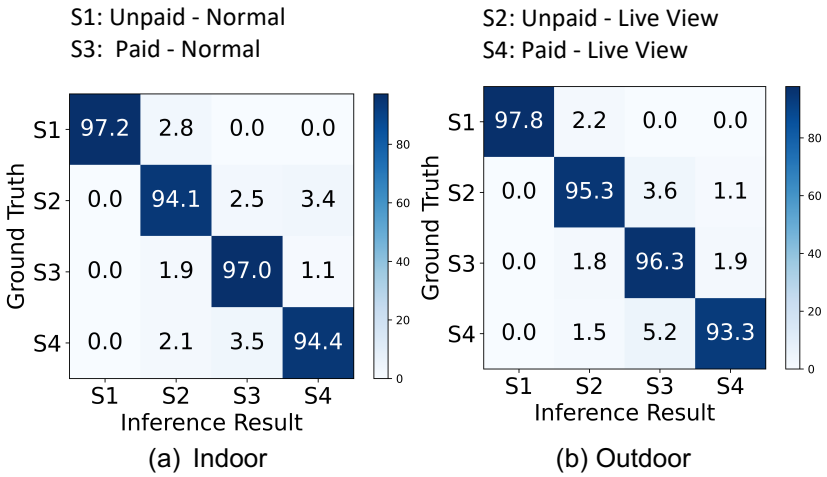


Figure 3.27: Overall confusion matrix.

Figure 3.27 plots confusion matrices of the inference results. We see that *WeakCamID* has consistently high true positive rates (93.3% or above) and low false positive rates (below 2.9%). We compute the F1 scores, which are both 0.96 on average for the indoor and outdoor environments.

Figure 3.28 plots the empirical cumulative distribution functions (CDFs) of the detection time T_{indoor} and $T_{outdoor}$ under the indoor and outdoor environments. We see no apparent difference in detection time for both environments. T_{indoor}

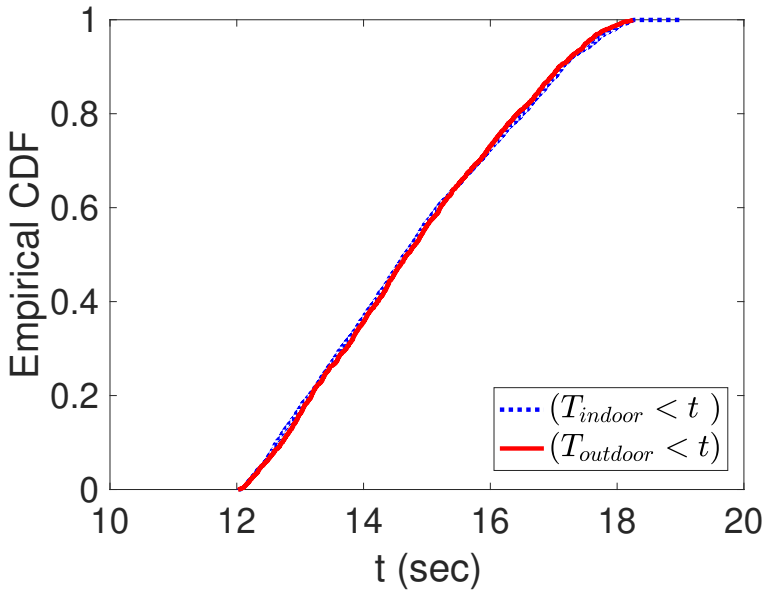


Figure 3.28: CDFs of detection time.

and $T_{outdoor}$ are less than 17.6 and 17.5 seconds with probability 95.0%. These results convincingly demonstrate that *WeakCamID* can effectively and efficiently infer camera states.

To further enhance inference accuracy, we evaluate the performance of RNN-based classification across all 13 cameras in both environments. Figures 3.25 and 3.26 present the success rates for the indoor and outdoor settings, respectively. Compared to SVM, RNN achieves improved classification performance, with success rates ranging from 95% to 99% indoors and 92% to 99% outdoors. This improvement is attributed to RNN's ability to capture sequential dependencies in traffic patterns.

However, this improvement comes at the cost of increased computational overhead. Figure 3.29 shows the empirical cumulative distribution function (CDF) of the detection time for RNN. Compared to SVM, the detection time distribution shifts slightly, with T_{indoor} and $T_{outdoor}$ now ranging between 15.7 and 19.2

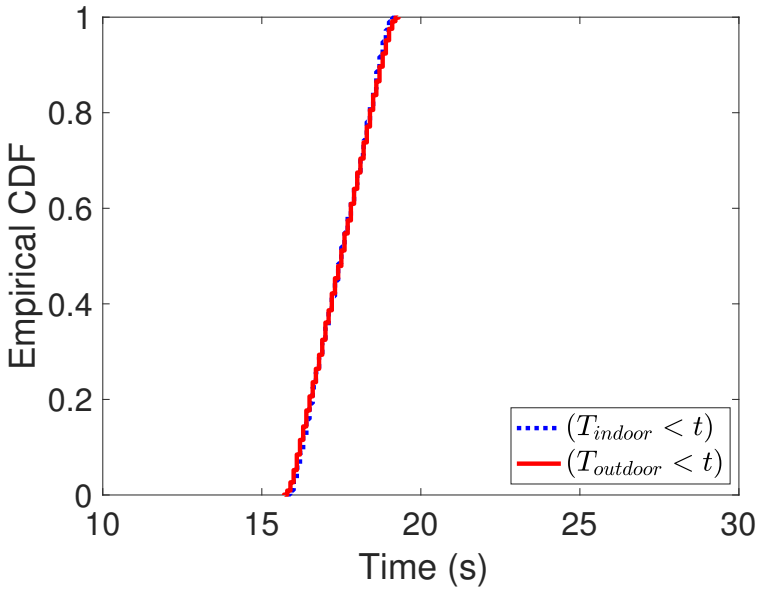


Figure 3.29: CDFs of detection time for RNN.

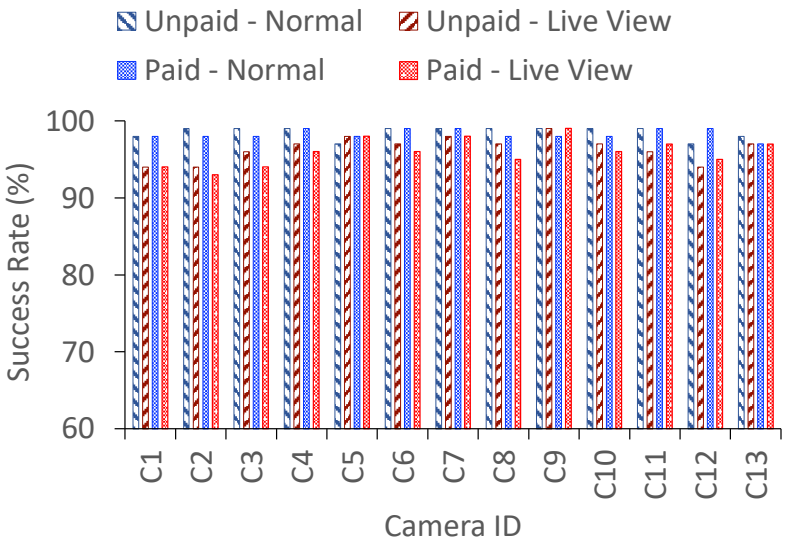


Figure 3.30: Indoor success rates for RNN.

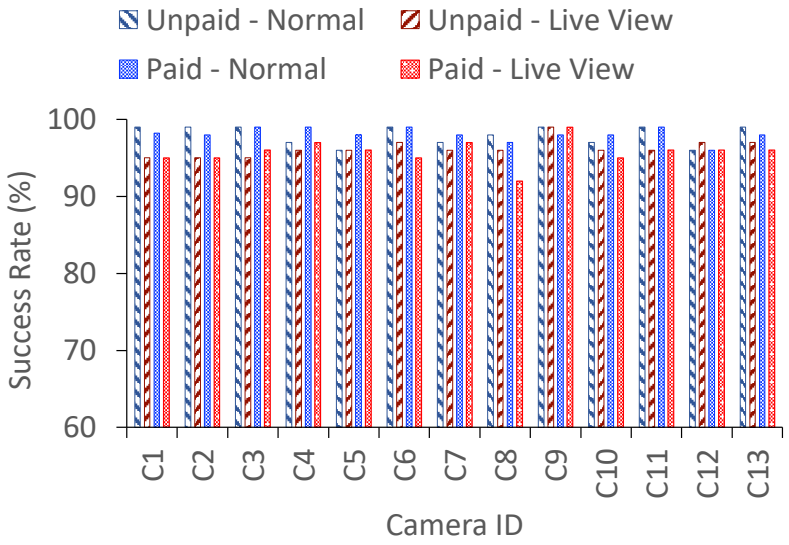


Figure 3.31: Outdoor success rates for RNN.

seconds, reflecting the higher computational complexity of deploying deep learning models in real-time classification. Despite this increase, the overall detection time remains within an acceptable range for practical deployment. These results demonstrate that while RNN significantly improves accuracy, its higher inference cost should be carefully considered when designing efficient camera state detection systems.

Occasionally, multiple wireless cameras may simultaneously monitor a target area. We present the corresponding evaluation in Section 3.5.8, verifying *WeakCamID* still works in such a scenario.

3.5.8 Multiple-camera Scenario

In a multi-camera scenario, the adversary needs to infer the states of all cameras in order to determine whether there is a risk of being recorded when performing motion in the area. *WeakCamID* tracks the wireless traffic based on MAC ad-

dresses. It can monitor multiple camera-associated traffic flows at the same time. Different cameras have no interference with each other for camera state inference.

To evaluate *WeakCamID* on a multiple-camera scenario, we deploy varying numbers of cameras (1 to 6) in the testing room. We manually tweak the fields of view of the cameras and make them overlap partially. We perform *WeakCamID* for 50 attempts for each camera count. We randomly change the location and state of each camera at every attempt. As the inference error mainly comes from current wireless traffic patterns and varies with the duration of performed motion, it is thus quite consistent across coexisting cameras. We find that for each camera count from 2 to 6, *WeakCamID* always successfully infers the states of all cameras with a probability exceeding 94.5%, similar to what we achieve for inferring a single camera's state.

Table 3.3 presents the mean, minimum, and maximum detection time of successful trials for different numbers of cameras. We find that when the camera count is no more than 3, the detection time just slightly increases with the count in most cases. This is because the one-time motion (i.e., walking) triggers all cameras at the same time and *WeakCamID* can take advantage of it to infer the states of all cameras. Thus, an extra camera only adds data processing time. Also, we see that when the camera count exceeds 3, it is often not enough to walk one time to trigger all cameras, and we have to perform several movements instead. As a result, inferring the states of multiple cameras is equivalent to inferring the state of a single camera several times, and the detection time is almost proportional to the corresponding number of performing movements. Overall, *WeakCamID* can infer the states of up to 6 cameras within less than three-quarters of a minute, demonstrating the high efficiency of the proposed technique.

Table 3.3: Detection time vs. camera count.

Camera count	Detection time (seconds)		
	Average	Minimum	Maximum
1	14.6	12.5	17.3
2	16.5	14.7	26.2
3	19.7	16.5	27.1
4	24.1	18.3	29.9
5	35.1	32.7	39.3
6	36.8	34.9	43.8

3.5.9 User Study

We recruited 11 volunteers (U1-U11; 5 self-identified as females and 6 as males) and asked each to perform *WeakCamID* to infer the state of a randomly selected camera deployed in the aforementioned indoor and outdoor environments.[2] Every participant performed 50 attempts for each camera state under each environment, and thus $50 \times 4 \times 2 = 400$ attempts in total. For each participant, the camera state appears in random order. Based on empirical results, we instructed the participants to introduce motion stimulation lasting 12 seconds or longer to achieve higher inference accuracy.

Figures 3.32 and 3.33 present the obtained success rates and F1 scores. We see that the average success rate and F1 score range from 91.0% to 95.0% and from 0.92 to 0.95, respectively. Also, regardless of the subscription status, the success rate or F1 score for the normal state is slightly higher than the live view mode.

Specifically, the average success rates of the states *Unpaid - Normal* and *Paid - Normal* for all users are 95.0% and 94.9%, while that for the states *Unpaid - Live View* and *Paid - Live View* are just 90.8% and 91.1%. These results convincingly

[2]The study has been reviewed and approved by our institution's IRB.

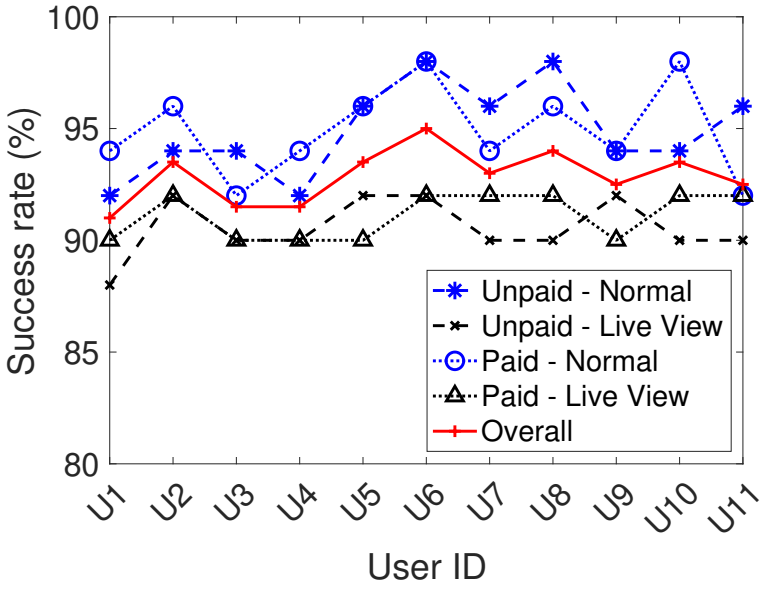


Figure 3.32: Individual success rates.

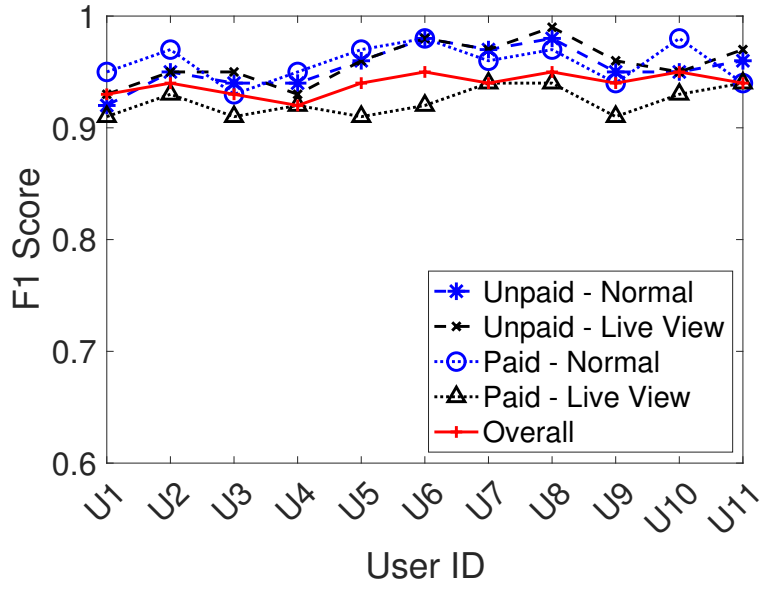


Figure 3.33: Individual F1 scores.

demonstrate that the performance of *WeakCamID* is robust to different camera states and users.

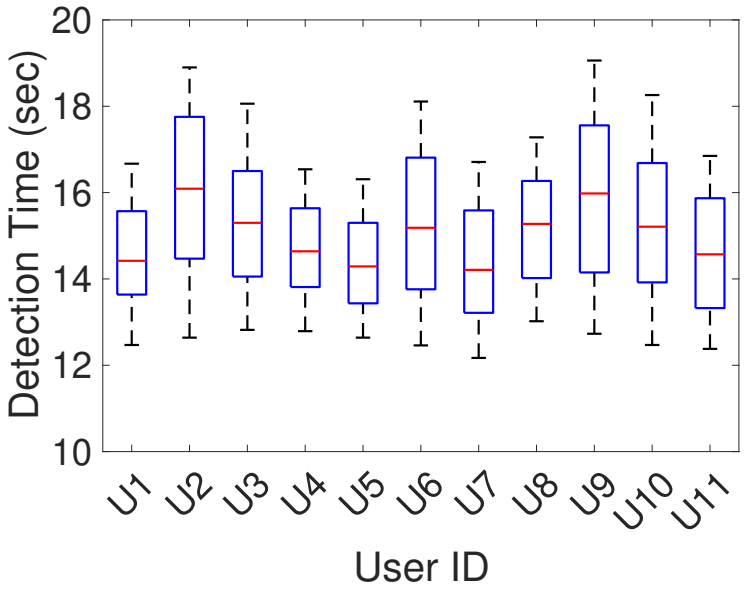


Figure 3.34: Individual detection time.

Figure 3.34 plots the users' detection time. We observe a consistent average detection time for all users varying from 14.3 and 16.0 seconds, indicating that a user can generally identify the camera state within a short period. This verifies the practicality of *WeakCamID*.

3.6 Discussions

3.6.1 Limitations

WiFi Dependency: *WeakCamID* currently only works for cameras using WiFi connections as it cannot scan cellular networks (e.g., 4G LTE or 5G) over a licensed spectrum. In reality, some cameras (e.g., Arlo Go 2 [32]) use cellular data, especially in locations without a dedicated Internet. The general *WeakCamID*

technique is expected to still work if it can sniff cellular traffic by incorporating special equipment (such as Universal Software Radio Peripheral) to receive signals in the corresponding carrier frequency range [105, 159].

Motion Requirement: To trigger wireless cameras, *WeakCamID* needs to generate motion stimuli, which may introduce potential risks of getting caught for the adversary. However, the adversary can perform normal and daily activities (e.g., walking) in disguise. Thus, even when such benign activities are recorded, they would not arouse suspicion and rarely harm the adversary. Alternatively, the attacker can ask a helper or control a moving object (e.g., a drone/robot) to perform movements for triggering the cameras.

Cameras with Local Storage: Some cameras or their base stations equip USB ports or microSD card slots for expandable local storage, which keeps recordings on local devices without sending them to the cloud. On one hand, if the local storage is enabled at the base station side (e.g., [3]), *WeakCamID* still works as there is still real-time wireless traffic between the camera and the base station for video footage. On the other hand, the local storage enabled on the camera side usually offers video storage when the WiFi goes out, such as Nest Cam [72]. In such cases, *WeakCamID* fails as the camera generates no real-time wireless traffic. It continues to work if the WiFi is on. Nevertheless, the owner of the camera often cannot view local storage via the app and has to connect it to a computer. Another downside to local storage is that if someone breaks in and steals the storage device (or the whole camera/base station), all video recordings on it would be lost.

3.6.2 Defending Strategies

WeakCamID exploits the correlation between the created motion and the resultant wireless traffic to infer camera states. Intuitively, to defend against such attacks, we can stop the attacker from obtaining the correct motion-traffic relationship. Accordingly, one straightforward defense is to disable motion detection of the camera, which will thus not react to any motion. To avoid losing the protection of the target area, the camera can then enable continuous video recording (CVR) that records videos continuously (24 hours a day). However, this feature can be unlocked via paid CVR subscriptions. Also, constant recording is power-consuming, making it often not supported by battery-power cameras. Instead, to support CVR, cameras are usually required to be plugged into AC power (e.g., [35]), bringing inconvenience to installation.

Alternatively, the camera can confuse the attacker by randomizing the time of uploading recorded videos. Currently, after sending the motion alert, the camera with a subscription also immediately records to the cloud. With this defense, the paying camera can postpone uploading recordings for a random delay, leading to two phenomena that increase attack difficulty. First, the attacker would only observe short wireless traffic for both paying and non-paying cameras right after performing motion in the activation zone of the camera. The corresponding traffic would be similar regardless of subscription status. Second, there is a break between the traffic flows for video uploading and for sending a push notification. Thus, the traffic for cloud recording can be confused with that for streaming live videos. As a result, the attacker is unable to correctly infer camera states. However, this defense needs local storage at the camera side to temporarily hold the recordings.

More expediently, we can disrupt the wireless traffic patterns observed at the sniffer by padding traffic (i.e., adding traffic volume) generated by the camera. Recent studies [25, 48] have demonstrated success in applying traffic padding to reshape traffic patterns of IoT devices. Particularly, two defense strategies can be taken, i.e., *obfuscation* and *deception*. Towards obfuscation, we pad the motion-induced traffic to make it featureless so that the traffic shows a consistent pattern regardless of the camera state. In terms of deception, we intentionally generate bogus traffic flows to mislead the attacker to infer a fake camera state. However, the defense inevitably imposes communication and power overheads.

3.6.3 Camera Models Not Present in Training

Currently, our implementation and evaluation are based on testing the 11 selected wireless cameras. It is difficult to enumerate all wireless camera models and test the corresponding performance of *WeakCamID*. However, Section 3.5.6 shows that even if one of the chosen 13 cameras is not used for training, *WeakCamID* still works for this camera with high accuracy and within a short time. Such results show preliminary evidence of the transferability of *WeakCamID*. In general, subscriptions have become a popular revenue model for camera vendors, and *WeakCamID* works for all cameras that have no active recording without a subscription.

3.7 Related Work

Detection and Localization of Wireless Cameras: Nowadays, the fast adoption of wireless cameras in miscellaneous venues has boosted the privacy research associated with unauthorized recording. A malicious user may stealthily deploy a

wireless camera and monitor an area without obtaining the approval of the occupants in the area. A myriad of recent studies ([44, 45, 86, 114, 152, 162, 189]) thus focus on detecting or pinpointing wireless cameras to protect user privacy. For example, [152] detects and localizes wireless cameras using the time-of-flight (ToF) sensor on commodity smartphones. Also, by exploring the causality between patterns in observable wireless traffic and a coexisting trusted sensor such as a light sensor [114], motion sensor [86], or inertial measurement unit (IMU) [162], a wireless camera can be detected or localized. On the contrary, our work is the first to explore how to infer whether a wireless security camera has a subscription plan without hacking into the camera or connecting to the same network with the camera.

Camera Jammer: Wireless communication systems are vulnerable to jamming attacks [57]. A malicious user may utilize a WiFi signal blocker or jammer to send interference using the same radio frequency as the camera [74], so that the traffic generated by the camera will be disrupted. However, such suppression methods not only need extra high cost for jamming hardware, but also disable all WiFi connections nearby. Also, as the heartbeat signals (reporting the camera's connectivity health to the server) will be jammed, the camera app will display an offline status message and an alert accordingly. Alternatively, the study [136] selectively suppresses sensory measurements (e.g., video recording) at the network layer rather than heartbeat messages. Nevertheless, this method requires pre-compromising the wireless router that the target camera connects to. Such a requirement imposes a practical hurdle for the attacker as the target camera usually utilizes a password-protected WiFi network that is not accessible to the attacker. On the contrary, *WeakCamID* is non-invasive, having no impact on environmental WiFi devices, and requiring neither expensive hardware nor control

over the router that the target camera connects to.

Traffic Analysis: Traditional traffic analysis is to extract and infer information from network meta-data such as the volumes and timing of network packets, even when the network traffic is encrypted [131]. In past years, traffic analysis has shown success in achieving a wide variety of applications, from classifying the traffic's nature (e.g., video streaming and p2p traffic) [140] to revealing fine-grained App usage [139, 170] or even user actions on Apps [16, 47, 176], and from unveiling streamed YouTube videos [111] to identifying IoT devices including drones [20, 135, 156], wireless cameras [44, 86, 162], and sound-triggered devices [129].

Most existing traffic analysis studies (e.g., [44, 111, 156, 170, 176]) take advantage of the fact that different applications or devices generate distinguishable traffic patterns or signatures, which in turn can be used to infer applications or devices. However, to determine the subscription and live-view mode on/off states, our work exploits the correlation between the manipulated motion stimuli fed into a wireless camera with the resultant traffic generated by the camera.

3.8 Conclusion

We propose WeakCamID for a novel and universal camera state inference technique. It is the first to point out the vulnerability of current wireless security cameras without subscription plans. An adversary may bypass such a camera without being recorded via passive WiFi sniffing. WeakCamID can be realized with a single smartphone and requires neither any professional equipment nor connecting to the same network as the target camera. It works by generating motion to stimulate the camera, and correlating the camera state (i.e., the

statuses of subscription and live view mode) with the disclosed traffic pattern. We also systematically evaluate both classical machine learning (SVMs, Random Forests, Decision Trees) and deep learning models (ANNs, CNNs, RNNs), finding that SVMs provide an optimal accuracy-efficiency tradeoff at 95% accuracy with 8.6 ms inference, while RNNs achieve 99% accuracy at higher computational cost. We develop a mobile app to implement WeakCamID. Extensive real-world experiments on top of the developed app and 13popular wireless cameras verify the effectiveness and efficiency of WeakCamID.

Chapter 4

Laser-Based Motion Injection Attacks on Wireless Security Surveillance System

This chapter [1] presents *PhantomMotion*, a new attack framework to fool the motion detection function of wireless security systems.

4.1 Introduction

Common wired security surveillance systems require a wired connection from the central hub all the way to each camera endpoint, at high cost and deployment time, especially if retrofitting a property already constructed. Traditional surveillance systems also require security guards with eyes glued to Closed Circuit Television (CCTV) arrays to identify human activities from these camera

[1] This work is based on *PhantomMotion*, published in NDSS 2026. I am authorized to reuse this paper and we extended the work with more experiments

feeds, which is quite inefficient and impractical on the small scale for private residences. In contrast, wireless surveillance systems have made property security more accessible and are widely deployed in smart homes and buildings, as they are easy to install and increasingly affordable [50]. Passive Infrared (PIR) motion sensors are often integrated with such systems and play vital roles in providing security. They automate device controls (e.g., videotaping detected activity and generating intrusion alarms) for an energy-efficient and safe home/office. According to a recent report, the global wireless video surveillance market was valued at 21 billion US dollars in 2021 and is projected to reach 64.1 billion US dollars by 2031 [19]. Also, it is forecasted that the global motion sensor market will increase at an annualized average growth rate of 12.3% from 2023 to 2033, and top a market size of 2 billion US dollars by 2033 [55].

The availability of wireless security systems is a two-edged sword. On one side, such devices improve the safety of our property and family by monitoring and reporting trespassing or other unauthorized activity [64, 142]. On the other side, they may be used for unauthorized tracking or video recording [106, 118], violating individuals' privacy. Given their increasingly smaller size and use of wireless transmission, the cameras can be easily kept hidden and even embedded within other objects, such as electrical appliances. The spy camera epidemic problem, targeted mostly at women and girls [173], has been a pressing issue in certain countries. For example, in South Korea, more than 30,000 cases of surreptitious filming with hidden cameras were reported to the police between 2013 and 2018 [38, 122]. People rightly value their privacy and do not wish to be recorded in secret, and similarly, malicious individuals such as burglars also do not wish to be recorded.

In both of these cases, it is valuable to determine the existence of such cam-

eras or pinpoint them, whether for the innocent individual seeking to protect their privacy or the malicious individual seeking to get away with a crime. There are thus emerging research efforts to perform wireless camera detection or localization [44, 45, 86, 91, 151, 160, 162]. Nevertheless, all of these studies share a common weakness, the requirement to perform human motion in front of the camera. Not all users are willing to perform preset motion events, especially a significant motion such as jump [44, 45]. Also, in case the user is suffering from physical disabilities, he/she may find it difficult to follow through the required motion schedules [162]. Furthermore, real human movements may inevitably disclose the person performing them. The motion-based detection or localization may identify an attacker aiming to find the camera and bypass it, defeating the purpose of finding the camera. Similarly, for victims aiming to detect unauthorized hidden cameras, the required human motion may harm their privacy to some extent. For example, motion-induced push notifications and video recordings may disclose the victim's presence, even if the victim does not perform private behaviors on camera. Theoretically, both types of individuals may act in disguise or hire a helper to perform motion. However, in practice, such solutions may cause considerable discomfort or inconvenience, and they still leave evidence for user tracking. Hence, we are motivated to investigate the feasibility of triggering wireless systems without labor-intensive or pre-designed human motion in proximity to the target systems.

Embedded PIR sensors in wireless systems convert received infrared radiation into a voltage. If the voltage exceeds a pre-defined threshold, the surveillance system will be triggered. We then explore how to inject radiation similar to what humans generate and fool the target PIR sensor. In this paper, we describe how to utilize a laser, a narrow beam of concentrated light, as the attack signal. Our

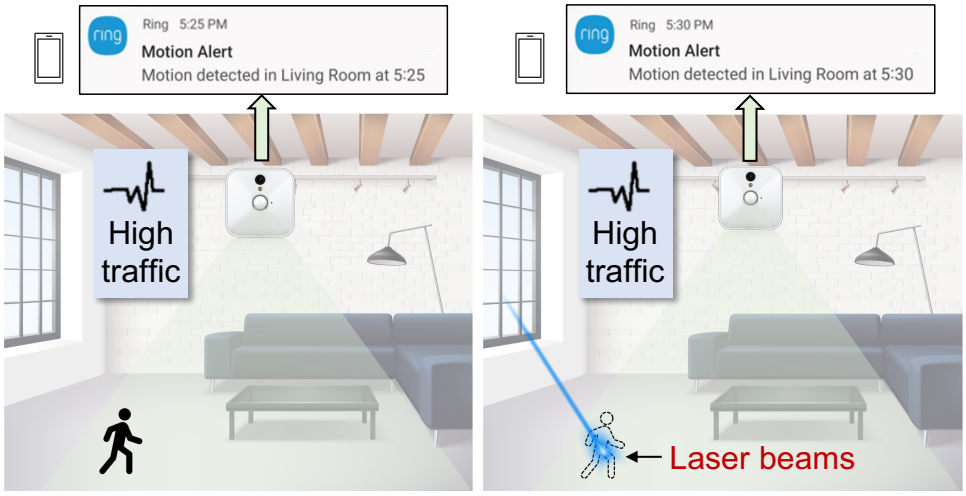


Figure 4.1: Creating phantom motion to trigger cameras.

key idea comes from the fact that lasers can rapidly heat materials via energy transfer [143]. By controlling the laser beams to simulate human motion in the detection area, an adversary can generate fake motion signals to trigger motion-activated wireless security systems, even with a long attack distance. We refer to the proposed attack technique as *PhantomMotion*.

We utilize a sample wireless camera as an example to illustrate how *PhantomMotion* works. Figure 4.1 (a) shows a scenario with real human motion, where a user comes inside the camera's motion detection range. The camera sits in standby mode in static environments. Upon detecting motion, it immediately turns on, starts recording to the cloud, and sends a push notification to the property owner's phone. Accordingly, the network exhibits sudden high wireless traffic.

Now consider the scenario in Figure 4.1 (b): there is no human motion in the camera's motion detection area, but the attacker wants to fool the camera into believing that motion similar to Figure 4.1 (a) is occurring. To this end, the attacker shoots a laser beam into the motion detection area through the window.

The laser is controlled to heat the chosen position for a period until a desired temperature (i.e., around normal body temperature) is reached. As a result, the radiation captured by the PIR sensor embedded in the camera becomes similar to that of a human, triggering the camera, which then generates high traffic and sends out a motion alert. In practice, the camera may be hidden, and the attacker may not foreknow where the exact motion detection range is or if the laser is within it. We accordingly design a customized laser scanning method, which first plots a route consisting of evenly distributed points and then determines whether the current laser point lies in the motion detection zone by correlating observed wireless traffic and laser heating time at the point.

Another important challenge is how to control the laser heating process to generate appropriate radiation at each laser point to activate the camera. Heating for a random duration may produce either insufficient or excessive radiation. The former will not trigger the camera at all, while the latter will make the relationship between laser heating and the camera's wireless traffic difficult to distinguish correctly. Our research reveals that the attacker can achieve laser heating control by reverse-engineering the motion detection mechanisms of PIR sensors and calculating the laser heating time accordingly.

Beside these issues, there are a variety of wireless traffic flows generated by different devices. *PhantomMotion* sniffs and collects encrypted wireless packets over the air to detect the motion-activated wireless system. We first coarsely detect the device type by inspecting the Media Access Control (MAC) addresses of captured packets. A MAC address is a persistent globally unique identifier. We can thus flag suspicious traffic flows generated by wireless security systems. By controlling laser heating, we can achieve more fine-grained detection of the security system that monitors the target area.

Unlike existing laser-involved research efforts (e.g., [166, 191]), which require users to manually operate laser beams, we develop hardware/software prototypes that enable the user to operate entirely through a non-rooted smartphone, making *PhantomMotion* easily accessible to non-technical users. We conduct a real-world evaluation, showing that an attacker can always fool the PIR sensors to activate the wireless system by injecting motion via laser heating, with a mean time of 12.8 seconds. We also verify that *PhantomMotion* can work at a distance of 120 meters, and even when the laser beams are within the non-line-of-sight (NLOS) of the target system.

In summary, we mainly make the following contributions.

- We propose *PhantomMotion*, the first practical method to activate wireless systems without requiring physical motion in close proximity to them. It can be carried out with a smartphone and needs neither professional equipment nor access to the target system's WiFi network.

- We discover an inherent vulnerability of PIR sensors embedded in wireless systems that can be exploited by laser to create a controlled heated position in their motion detection zones, injecting fake motion signals.

- By taking advantage of how the motion detection range and the field of view of the camera's optics are not exactly the same, we show how a visible laser triggers the system without being recorded by the camera. We also show how *PhantomMotion* stealthily works with a wall or other obstacles between the camera and the laser.

- We build a low-cost hardware platform with off-the-shelf sensors and develop an app to automate *PhantomMotion* and validate its feasibility, effi-

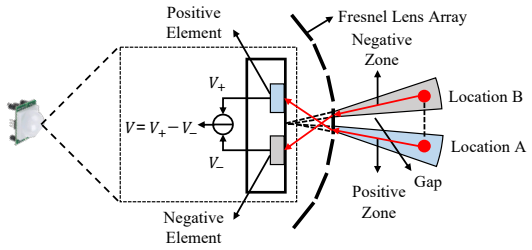


Figure 4.2: Structure of a PIR sensor.

ciency, and robustness.

Laser Safety: Laser stands for Light Amplification by Stimulated Emission of Radiation. Laser radiation requires special controls for safety. The American National Standards Institute classifies lasers according to their relative hazard [108], as shown in Section 4.10. All experiments in this paper were conducted under a Standard Procedure reviewed and approved by our institutional Office of Compliance.

4.2 Principle of Motion Detection

There are multiple types of motion sensors, such as Passive Infrared (PIR), ultrasonic, microwave, and tomographic. Among them, PIR sensors are widely utilized in wireless security systems, due to their small size, low price, high sensitivity to motion, and ability to work even in dark environments.

A PIR sensor contains a pyroelectric sensing element, which is sensitive to radiated heat power fluctuation and produces voltage output when exposed to heat in the form of infrared radiation (IR) [71]. Figure 4.2 shows the general structure of a PIR sensor. A Fresnel lens array, which condenses light, can provide a larger range of IR and focus it to a small point (i.e., the focal point where the pyroelectric sensing element is mounted) [133]. When a warm body like a person moves from Location A to B, the IR emitted from the body, denoted as the red

dot, passes through the Fresnel lens array. As a result, when the person is at Location A, the IR first only intercepts one half of the PIR sensor (i.e., positive element), causing a positive differential change between the two halves; likewise, when the person moves to Location B, only the other half of the PIR sensor (i.e., negative element) recognizes the IR, leading to a negative differential change.

The two elements of the PIR sensor generate a respective output voltage, denoted as V_+ and V_-, and the final output of the sensor V equals $V_+ - V_-$. The absolute value of V is then quantized based on a pre-defined threshold V_0 to generate a binary event. Specifically, if the detected heat by the PIR sensor causes V to exceed V_0, motion is detected; otherwise, no human presence is detected.

Heat Signature: PIR sensors detect the heat (infrared) signature of an object, which can be represented by the physical temperature difference between the object and the background (referred to as ΔT) [51]. According to Stefan-Boltzmann law [154], the power density P, i.e., the energy in watts (W) per unit surface area in unit time, of electromagnetic radiation emitted by a black object (i.e., a hypothetical physical body absorbing all incident electromagnetic radiation) is directly proportional to the fourth power of its absolute temperature. Let Q denote the total generated heat when the exposure time of the object is t. Mathematically, we then have

$$Q = P \cdot t = (\Delta T)^4 \cdot \sigma \cdot s \cdot t, \tag{4.1}$$

where s is the size of the considered surface (in square meters), and σ is the Stefan-Boltzmann constant and equal to 5.67×10^{-8} W/m^{2}K^4. The SI (International System of Units) [137] unit for ΔT is the kelvin (i.e., K for short).

4.3 Adversary Model

For the remainder of this paper, we define the "adversary" as in individual (victim seeking privacy or malicious person avoiding authority) who is interested in triggering the target wireless motion-activated camera without being detected. Towards the goal, the adversary uses a laser to stealthily simulate human motion, tricking the camera to generate motion alerts when there is no human present in the monitoring area. By triggering the camera and monitoring the resultant wireless traffic, the adversary can determine whether a camera is monitoring the area. Before taking malicious behaviors, such as burglarizing or invading a home, or just to protect one's privacy, the adversary can even drain the battery of the camera by consecutively activating it. In this case, the camera will be activated by the manipulated and fake human motion, but it will record no event. Such motion alerts are inaccurate or irrelevant, and may make people disable notifications or become desensitized to them [78]. There are many users complain with false alarm at false alarm notification especially during the night, when it will be annoy for them [31, 34].

Unlike all existing studies (e.g., [44, 45, 86, 162]) requiring human intervention (e.g., waving hands, walking around, or jumping) within the proximity of the cameras to activate them, we assume that that the adversary is out of the camera's field of view, with remote line-of-sight access to the target monitoring area. Note that introducing real human motion is not an optimal strategy for the adversary as she or her accomplice (who performs motion) may get caught during this triggering process. We argue that our assumptions are more reasonable than those of the related work, as the adversary will prefer to stay hidden rather than intrude into the monitoring area and disclose her location/identity.

4.4 Injecting Motion via Laser Light

4.4.1 Feasibility Analysis of Motion Injection

Support that the human temperature T_h is 37 ℃, corresponding to 310 K. Note that the Celsius scale has an origin translated into 273.15 K as regards the Kelvin scale. A human body is not a perfect black body, while it is proved that the difference between the radiation emission characteristics of a human body and a black body is small [97]. Considering that the approximate size of a laser point is 1 cm × 1 cm, we regard the unit surface area $s = 10^{-4}$ m^2. Thus, according to Equation 4.1, the power density of the radiation emitted by the of human body with a unit surface area with background temperature of 20 degrees Celsius can be denoted by

$$P_h = (310^4 - 293^4) \times 5.67 \times 10^{-8} \times 10^{-4} = 0.01 \text{ W}. \tag{4.2}$$

To trigger the PIR sensor, a human may have to disclose at least U unit surface areas to PIR sensors to generate enough radiation. Let P_l denote the power of the utilized laser. When we have $P_l \geq P_h \cdot U$, i.e., the generated power of a laser point is equal to or higher than that generated by a human, the laser is able to trigger the PIR sensor with the generated radiation.

Empirical Verification: We utilize the two sources (i.e., laser and human) to simulate a PIR sensor (with model HC-SR501 [7]) and compare the resultant responses from the PIR sensor. We utilize a FLIR One Pro thermal camera [171] attached to a phone to capture the infrared radiation, which is invisible to the naked eye, and determine that the surface temperature of a walking person is 37

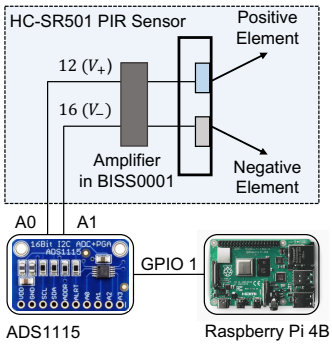


Figure 4.3: Setup for obtaining a PIR's raw voltage output.

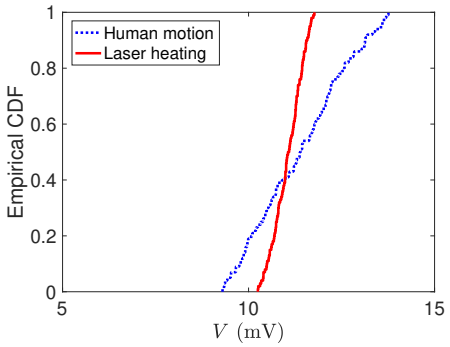


Figure 4.4: CDFs of the voltage output of a PIR sensor.

℃, which serves as a reference for the cut-off temperature of laser heating. As discussed in Section 4.2, the PIR sensor only provides a binary output, i.e., 0 (representing no motion) and 1 (indicating motion). The raw voltage output of a PIR sensor can provide fine-grained information about the captured radiation.

We disassemble the HC-SR501 PIR sensor and identify its controller BISS0001, which contains an operational amplifier to boost the motion-induced voltage to improve the reading accuracy [6]. As shown in Figure 4.3, we employ a Raspberry Pi to connect with an ADS1115 module [174] via GPIO 1, which is a 16-bit analog-to-digital converter (ADC); two input pins (A0 and A1) of the ADC are further connected with the two output pins (pins 12 and 16) of the module BISS0001, respectively. As a result, the Raspberry Pi can intercept the amplified voltage readings V_+ and V_-, and calculate the PIR's final voltage output $V = V_+ - V_-$.

We let human motion (walking) trigger the PIR sensor. Meanwhile, when

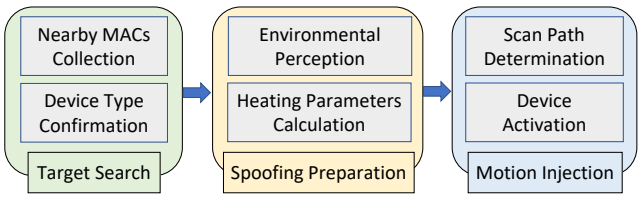


Figure 4.5: Three core phases of the proposed scheme.

no motion occurs, we use a laser to heat the place where the user walks. We monitor the real-time temperature of the heating point via the thermal camera and cut off the heating once the temperature reaches the reference (i.e., 37 ℃). For both human motion and laser heating, we perform 100 trials. Likewise, we see that all laser heating attempts succeed to activate the PIR sensor. Figure 4.4 plots the empirical cumulative distribution functions (CDFs) of V under different scenarios. We observe that the generated voltage for human motion ranges from 9 to 14 mV, while laser heating induces a comparable voltage, varying from 10 to 12 mV. These results verify the possibility of using a laser to mimic human motion for triggering PIR sensors.

4.4.2 System Overview

The proposed motion injection attack consists of three core phases, *target search*, *spoofing preparation*, and *motion injection*, as illustrated in Figure 4.5.

The initial phase obtains the wireless traffic associated with wireless surveillance devices. The attacker first captures the wireless 802.11 traffic flows over the air, and groups the collected packets based on their embedded MAC addresses. The device type for each scanned MAC can be predicted accordingly. After flagging the traffic flows belonging to potential target wireless security systems, the attacker then enters the second phase, which consists of two sequential preparatory tasks: (i) to perceive key environmental factors (such as the temperature

and heating material); and (ii) to determine the corresponding parameters (e.g., heating time at each spot) for laser heating. Finally, *PhantomMotion* monitors traffic flows flagged in the first phase, and meanwhile initializes the laser heating along the chosen scan path to inject fake motion into the target wireless security system. By correlating the laser heating time with the wireless traffic that the monitored security system generates, *PhantomMotion* can figure out whether the target system is triggered or not. We present the details of *PhantomMotion*'s design in the following discussion.

4.4.3 Target Search

The target motion-activated device can connect to a password-protected WiFi network, and thus it may be present in any possible WiFi channel. Meanwhile, there are diverse wireless traffic flows in the air. Thus, *PhantomMotion* first needs to determine the traffic flow belonging to the target device. We identify the possible candidate of the target device based on the publicly available information of manufacturers embedded in captured MAC addresses.

4.4.3.1 Nearby MACs Collection

PhantomMotion cannot access the same WiFi with the target device, and thus needs to scan all possible channels that the target wireless security system may operate on. Wireless sniffing tools, such as Airmon-ng [5] that is open source, enable *monitor mode* (i.e., monitoring all wireless traffic nearby) on wireless interfaces. Modern Android smartphones with built-in WiFi chips that support monitor mode can work as traffic sniffers, while it is often required to first root the devices [169] and pre-install customized firmware (e.g., [10]).

IEEE 802.11 wireless protocols are used in almost all commodity network devices [120], including various wireless security cameras and alarm systems. Though WiFi networks employ security protocols (WEP, WPA, WPA2, and WPA3) to encrypt transmitted wireless data, 802.11 management frames are unencrypted, from which we can extract the MAC addresses of the devices that generate the corresponding packets. With a wireless interface in monitor mode, *PhantomMotion* can capture raw wireless packets and identify all active channels nearby. It can then collect all MACs and feed them to the next module for further winnowing out the traffic flows belonging to the target security system.

4.4.3.2 Device Type Confirmation

Wireless security systems rely on systems-on-a-chip (SoCs) from SoC manufacturers (such as Texas Instruments, Broadcom, and Qualcomm) to provide critical wireless communication capabilities. An SoC has a Media Access Control (MAC) address, which is a unique identifier assigned to its network interface controller (NIC). A MAC address consists of 6 bytes (48 bits) that are typically represented as 12 hexadecimal characters. The first 3 bytes are the Organizationally Unique Identifier (OUI), which identifies a manufacturer or a vendor, and the rest 3 bytes represent the unique device ID. For example, Ring LLC has three OUI IDs, i.e., "9C-76-13", "54-E0-19", and "34-3E-A4" [11], used for its products, including wireless cameras, doorbells, and security alarm systems.

PhantomMotion builds a library with public OUI IDs of wireless security systems and utilizes it to look up the device manufacturer for each collected MAC address. If a match is found, the traffic carrying the MAC is regarded as being generated by a wireless security system.

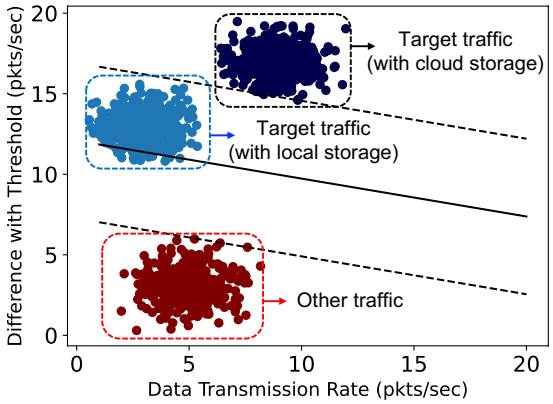


Figure 4.6: Traffic flow identification via the built SVM models.

MAC Spoofing: The MAC address can be spoofed or randomized. The studies [119, 120] propose to retrieve the device's original MAC with its Universally Unique IDentifier-Enrollee (UUID-E) that is used to establish WiFi Protected Setup (WPS) connections. Also, different types of devices may have varying traffic patterns [45], and we can take advantage of this observation to distinguish traffic flows belonging to wireless security systems.

The SoC chip of a wireless system provides data processing and encoding solutions. For example, a wireless camera's SoC (e.g., Ambarella's CV2S SoC [22]) often pre-determines the video encoding methods including H.264, H.265, and MJPEG. If the wireless system has local storage, it normally sends out event notifications and stores the recorded content locally.

We train two one-versus-one linear kernel SVMs using Scikit-learn [157] libraries with Python 3.8.1, to classify traffic flows belonging to wireless security systems and other devices. We set the hyperparameter c to 10 for error control of the wireless traffic. The training data split in an 80/20 ratio for training and validation. Different types of devices may have varying data transmission rates. We set a threshold based on the average data transmission rates of various wireless devices in the environment. For each traffic flow, we calculate the difference

between its data transmission rate and the threshold.

Figure 4.6 shows the results after training 2 SVM models on 400 traffic flows from each of the three classes: target device (i.e., wireless security system) with local storage, target device with cloud storage, and non-surveillance devices, demonstrating the success of the employed traffic flow identification technique.

4.4.4 Spoofing Preparation

4.4.4.1 Environmental Perception

In order to determine the laser heating time at a chosen point, we need to know the environmental temperature (denoted as T_e) and the solid heating material.

We utilize a temperature sensor (e.g., DHT-11 [2]) to capture T_e. As the material's temperature increases, the heat is stored in its molecules. When the material cools down, the stored heat is then released. Let T_t denote the target temperature that is able to trigger the motion-activated wireless security system. We aim to simulate human motion with laser heating, and thus set $T_t = 37$ ℃.

Different materials may have varying densities and specific heat capacities, which affect the amount of heat required to raise the temperature of a unit volume of a substance. For common building materials (such as wood, glass, iron, and steel), we first build a library containing different materials' densities and specific heat capacities. Such a library is then prepared for the next step to calculate the heating parameters.

4.4.4.2 Heating Parameters Calculation

Let Q denote the amount of heat required for the laser beam to increase the temperature of a substance by ΔT (i.e., $T_t - T_e$) with a volume v. Accordingly,

we have

$$Q = \rho \cdot c \cdot \upsilon \cdot \Delta T, \tag{4.3}$$

where ρ and c are the density and specific heat capacity of the target object, respectively [186]. The laser beam has a spot size of 1 cm $\times$ 1 cm and only needs to heat the surface of the material. Without loss of generality, we consider heating a thin layer with a thickness of 1 mm. Thus, we have $\upsilon = 0.1$ cm^3. When the heating power of the employed laser is P, we can then obtain the time t_{min} that it takes for the laser to generate the required heat, i.e.,

$$t_{min} = \frac{Q}{P} = \rho \cdot c \cdot \frac{\upsilon \cdot (T_t - T_e)}{P}. \tag{4.4}$$

For three popular building materials including wood, stone, and brick [52], we calculate t_{min} under varying environmental temperatures based on Equation 4.4. Meanwhile, we also perform real-world experiments to measure the required time for each material to reach the temperature of T_t with laser heating. Specifically, we utilize a FLIR One Pro thermal camera to monitor the temperature of the laser spot. We then cut off the laser once the temperature increases to T_t and record the laser heating time. We perform 20 trials for every environmental temperature and compute the average laser heating time. We use the pigpio library [9] to control a General Purpose Input Outputs (GPIO) port connected to the laser module, with a sampling rate between 100,000 and 1,000,000 times per second. As a result, we can achieve initiating/stopping the laser heating with the 10 microseconds level of time accuracy.

Figure 4.7 plots the required time for increasing each material's temperature to T_t with a 100 mW laser under different scenarios, as well as the empirically

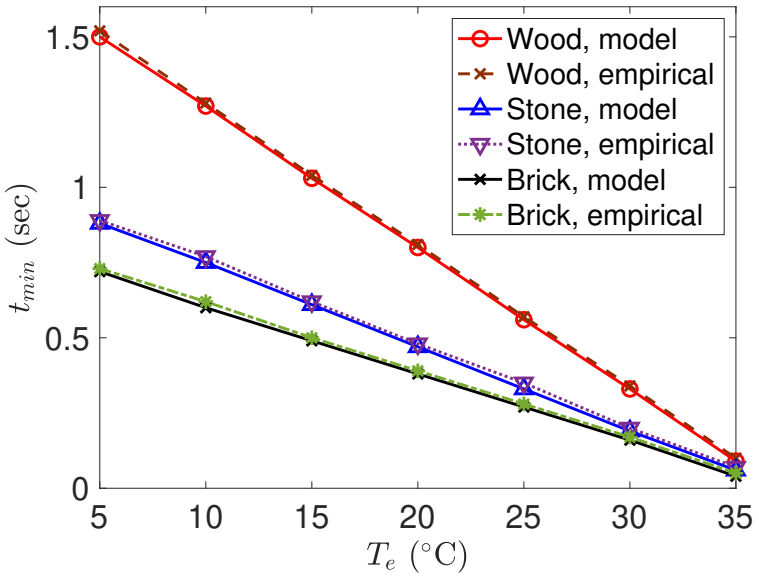


Figure 4.7: Required heating time.

obtained corresponding mean laser heating time. We see that the theoretical and empirical laser heating time values are quite consistent, and they are inversely proportional to the environmental temperature. Also, regardless of the environmental temperature, the required laser heating time is in the increasing order to brick, stone, and wood.

4.4.5 Motion Injection

A PIR sensor converts received infrared radiation to a voltage, which is then compared with a pre-defined threshold to generate a binary value indicating whether motion is detected or not [115]. Usually, the field of view (FOV) of the camera is no larger than that of its embedded PIR sensor. For example, a Ring floodlight camera has a FOV of 140° while the FOV of its PIR sensor is 270° [148].

We consider a wireless security camera deployed on a vertical wall. Such a case aligns with most practical scenarios. Meanwhile, in order to obtain the

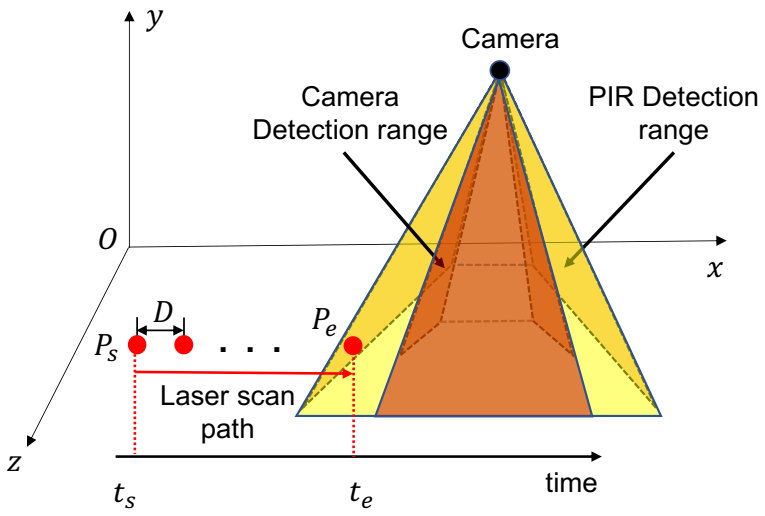


Figure 4.8: Outside-in scanning strategy.

maximum horizontal breadth, the camera body is often mounted perpendicular to the wall, and thus, the leftmost or rightmost areas may not be covered by the camera's FOV. Let L denote the distance between the laser and the wall.

To prevent the camera from recording the visible laser beams, the laser scan path should avoid the camera's FOV. Accordingly, we propose an *outside-in scanning* strategy to spoof motion via laser heating. As shown in Figure 4.8, the attacker performs laser heating along the path and simultaneously monitor the traffic, including the following steps,

1. Initially select a starting point P_s at the leftmost (or rightmost) area, and track the corresponding time t_s.

2. Heat the selected position to the pre-determined temperature for a period of t_{min}.

3. If the camera is not activated, move the laser point for a distance of D (referred to as the moving step) to the right (or left), and jump to step 2;

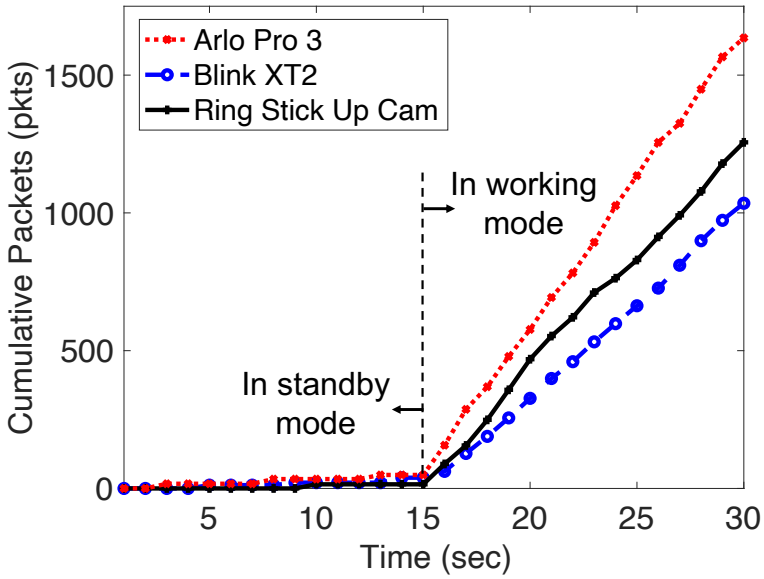


Figure 4.9: Cumulative packets.

otherwise, at time t_e, the ending location is marked as P_e, and record the moving angle of the laser beam as θ.

Note that in order to move the laser point for a distance of D, we need to turn the laser beam for an angle of about $\arcsin(\frac{D}{L})$. Suppose that the laser scans N locations along the path. Accordingly, the length S of the laser scan path can be computed as $S \approx L \cdot \sin\theta = N \cdot D$. For the total time T_o for the spoofed motion to activate the camera, we thus have $T_o = t_e - t_s = N \cdot t_{min} + (N-1) \cdot t_0$, where t_0 is the time spent for shifting the laser beam from one location to another. When the laser angle adjustment process is fast, t_0 becomes negligible, causing $T_o \approx N \cdot t_{min}$. Occasionally, a whole scan path is not within the target motion detection range, leading to the failure of triggering the camera at all locations along the initially chosen scan path. A new scan path can be then selected and the above three steps are repeated until the attacker succeeds to activate the camera.

System Activation Detection: When the wireless security system is triggered from the standby mode, it enters the work mode and responds accordingly (e.g., starting to record videos and send push notifications). From a wireless traffic perspective, the total wireless packets generated by the system would increase at a faster speed with the activation period. The large deviation between the packet generation rates in standby and working modes provides a clue to distinguish the two modes.

We also empirically verify such a phenomenon. Specifically, we install an Arlo Pro 3, a Blink XT2, and a Ring Stick Up Cam on a wall to monitor a target area. We perform a 30-second test for each device and collect its generated packets. For the beginning 15 seconds, the cameras are in standby mode for conserving battery, while we trigger the cameras for the rest 15 seconds. Figure 4.9 plots the count variation of the cumulative packets for the three cameras in different modes. In standby (low-power) mode, the packet count increases quite slowly as only a small amount of packets, i.e., periodical heartbeat signals, will be generated for synchronization with the server. On the contrary, in working mode, the count of cumulative packets increases with time at a much higher rate and they show a nearly linear correlation regardless of the camera type. For example, the Ring Stick Up Cam generates 15 packets before being activated, while the amount of the total generated packets reaches 1256 after a 15-second activation period, indicating a packet generation rate of about 83 packets per second.

The discovered correlation between the working duration (when the camera is activated) and total packet count can be explored to determine whether the camera is activated by the spoofed motion induced by laser heating, when the attacker is able to sniff wireless traffic and obtain the total packet count.

4.4.6 Exploring Stealthy Attacks

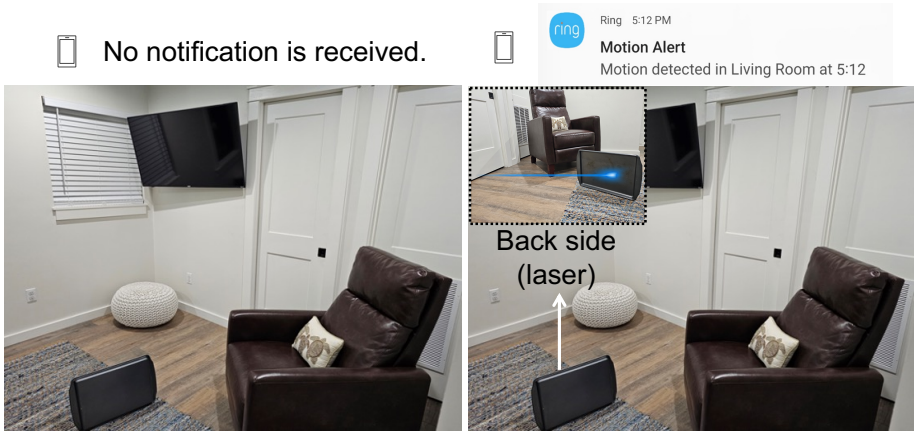
Occasionally, if the laser beam happens to appear in the FOV of the camera and triggers the camera, it may be spotted via the recordings. To further improve optical stealthiness, the attacker can use an invisible laser wavelength-] [46] to avoid having the user spot the laser light aimed at the target area.

Attack in Non-line-of-sight Scenarios: Furthermore, considering that heat (radiation) can be transferred over opaque obstacles/walls, *PhantomMotion* also works within non-light-sight (NLOS) of the target wireless system. In such an environment, the attack stealthiness can be guaranteed even when a visible laser is employed, as video recording requires line-of-sight (LoS) between the camera and the target. Specifically, we can utilize a solid daily object or wall as camouflage. One side of the object faces the camera while we shoot the laser beam to the opposite side of the object. The camera can be triggered once enough heat is transferred to the side facing it.

Let ΔT_0 denote the difference in temperature between the hot surface (heated by the laser beam) and the cold side (within the motion detection range of the camera). The distance between the hot and cold sides (i.e., the thickness of the material) is denoted as Δx. Accordingly, the quantity of heat energy transferred (denoted with q) can be obtained from Fourier's law [94],

$$\frac{q}{\Delta t} = -A \cdot k \cdot \frac{\Delta T_0}{\Delta x}, \tag{4.5}$$

where Δt is the time taken, A is the area of the surface that emits heat, and k is the thermal conductivity (i.e., a property indicating the ability to transfer heat). As the heat flows from the side at a higher temperature to the one at a lower



(a) Normal situation. (b) Launching the attack.

Figure 4.10: Camera feeds with no attack and the NLOS attack.

temperature, the heat transfer rate $\frac{q}{\Delta t}$ is always negative.

Meanwhile, during the heat transfer process, there may exist heat loss, causing a temperature drop ΔT_1, which can be modeled via Newton's Law of Cooling [187],

$$\Delta T_1 = \Delta T_0(1 - e^{-\frac{k}{\Delta x}t}), \tag{4.6}$$

where t signifies the elapsed time.

Let Q represent the heat generated by laser heating, which can be determined with Equation 4.3. The power of the laser is P and the laser heating time is t_{h_0}. Thus, we have $Q = P \cdot t_{h_0}$. With Equation 4.5, we can then calculate the time t_t needed for heat transfer, i.e., $t_t = \frac{Q}{A \cdot k \cdot \Delta T_0 / \Delta x}$. With obtained t_t and Equation 4.6, we can obtain the temperature drop ΔT_1 and further compute the laser reheating time t_{h_1} for compensating the heat loss. The total time needed for the heat produced and transferred to the target surface equals $t_{h_0} + t_{h_1} + t_t$. Figure 4.10 presents an example of triggering a camera by using a laser to heat the back side of an object whose front side faces the camera. We see that though the camera is triggered, its image capturing the scenario is exactly the same as that in the

normal situation (i.e., without the attack).

4.5 Experimental Evaluation

We build *PhantomMotion* with low-cost commercial devices, which can be easily purchased online. We also develop an Android app for controlling the hardware platform.

4.5.1 Evaluation Setup

The default mode for a wireless device's NIC is *managed mode*, in which the device only listens to the traffic sent to it and discards any packets not destined for it. To sniff all WiFi data, existing techniques usually utilize either certain models of laptops (such as Lenovo Thinkpad [162] and MacBook Pro [130]) or rooted Android platforms [86], which can enter monitor mode. However, not all laptops support monitor mode, and it may not be convenient to bring a bulky laptop when performing *PhantomMotion*. Also, rooting a smartphone is not easy for all users [163] and it may make the device vulnerable to malware and many other attacks [158].

Alternatively, we design a comprehensive, low-cost, and portable system that integrates WiFi sniffing and laser control. The system has the following four components, as shown in Figure 4.11: (1) a temperature sensor (e.g., DHT11) for measuring environmental temperature, and (2) a WiFi adapter (e.g., AWUS1900 [18]) in monitor mode, (3) a laser module (e.g., LD-F405E04 100 mW) with a laser level swivel base, and (4) a Raspberry Pi 4B controller interacting with other components via respective GPIO (general purpose input and output) and USB ports. The total cost of the whole hardware platform is around 80 US dollars.

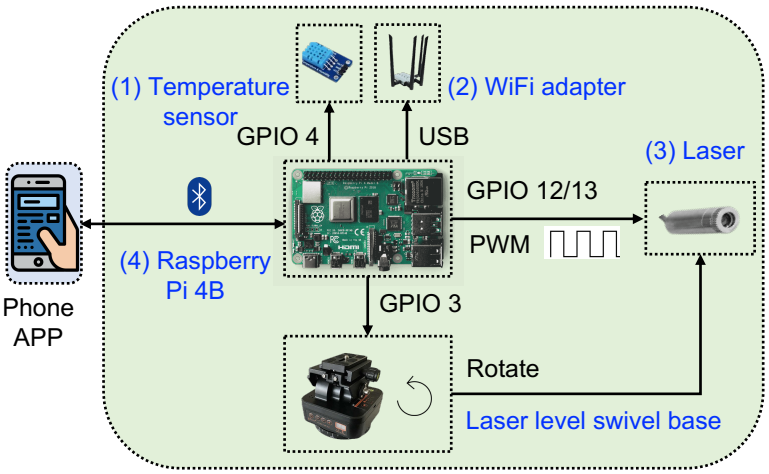


Figure 4.11: The testbed that we implement.

Specifically, the app connects to the Raspberry Pi board via Bluetooth Low Energy (BLE) to obtain system running information such as laser exposure time and environmental temperature. It can also send commands to the board to turn on/off the laser remotely as well as tilt or rotate the laser level swivel base.

We test 18 most popular wireless security devices, including 15 cameras and 3 alarm systems, as shown in Table 4.1. Among them, three cameras (ID 11-13, referred to as Type 2) have two PIR sensors while the rest devices (categorized as Type 1) just have one. For the testing scenario, we select a 400 square feet living room (20 ft * 20 ft), as shown in Figure 4.12. A wireless security camera or alarm system is mounted on the wall to monitor the room.

Metrics: We use the following three metrics.

- *Success Rate:* It is the ratio between the number of successful light sensor attack (i.e., activating the faked daylight mode) and the total number of attack trials.

- *Operation Time:* It is the total amount of time spent on triggering the

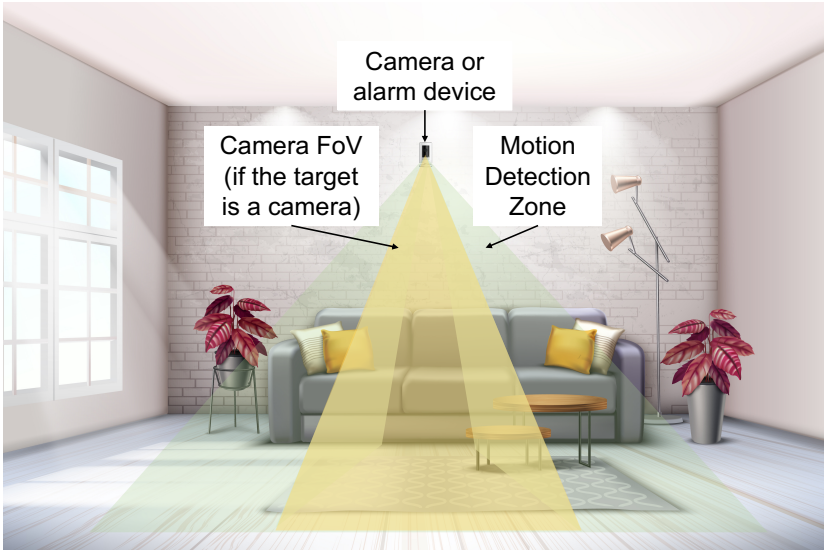


Figure 4.12: Experiment environment.

motion detection function of the target devices via the laser-generated fake motion.

- *Scan Distance:* This is measured as the distance between the initial laser point to the final one, at which the motion detection is triggered.

Safety measurement: Our experimental setup was located within a residential living room, with the explicit consent of the homeowner. During each experimental procedure, the room was secured with the door locked to ensure safety. All windows were adequately covered using laser safety curtains to prevent any inadvertent laser escape. Furthermore, the individual who is involved for conducting the experiment was equipped with laser safety goggles to provide additional protection.

Table 4.1: Tested wireless security devices (with respective model names in alphabetical order).

ID	Model	WiFi Chipset	PIR Amount
1	Arlo Pro 2	Cypress	1
2	Arlo Pro 3	Cypress	1
3	Blue by ADT	Cypress	1
4	Blink XT2	TI	1
5	eufyCam E	Hisilicon	1
6	Google Nest Cam	Ambarella	1
7	Google Nest Doorbell	Ambarella	1
8	IHOXTX DF22 Cam	MediaTek	1
9	LaView N15 Cam	MediaTek	1
10	Reolink Argus 2	MediaTek	1
11	Ring Spotlight	TI	2
12	Ring Spotlight Pro	TI	2
13	Ring Stick Up Cam	TI	2
14	Simplisafe Cam	Telit	1
15	Wyze Cam Outdoor v2	Ingenic	1
16	Arlo Home Security System	Cypress	1
17	Ring Alarm System	Quectel	1
18	Simplisafe Safety Alarm	Espressif	1

4.5.2 Case Study

In this example, a Ring Stick Up wireless camera is installed in the room, as shown in Figure 4.12. *PhantomMotion* is then launched 10 times. We change the camera's location to make the camera aim at different areas each time. The laser module is six meters away from the wall where the camera is. We set the moving step length D (i.e., the distance between two successive laser points) as 0.5 m.

Figure 4.13 depicts the nearby traffic flows. The user injects fake motion, and we observe a strong correlation between the camera traffic throughput and the laser heating. Specifically, once the camera is triggered, the count of newly generated packets matches the new heating time at the ending location of the laser beam. Meanwhile, the other two observed traffic flows do not have an obvious

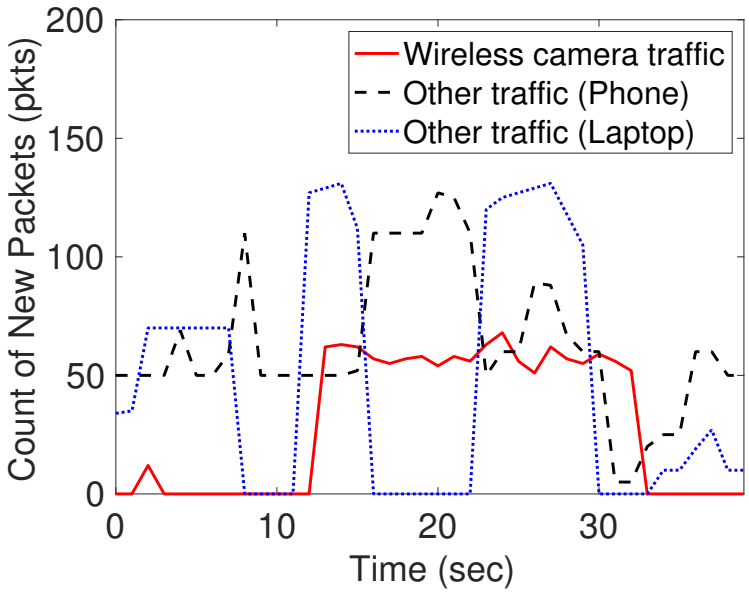


Figure 4.13: Traffic observation.

relationship with the phantom motion, and they belong to an iPhone in use (online chatting) and a MacBook Pro laptop in web browsing mode, respectively, via further examining their associated MAC addresses.

For all 10 tests, *PhantomMotion* successfully triggers the target camera, i.e., the success rate reaches 100%. The operation time and scan distance for each test are shown in Figures 4.14 and 4.15. We observe that the average operation time and scan distance are 2.9 s and 1.25 m, respectively, indicating the efficiency of *PhantomMotion*.

4.5.3 Influential Factors

4.5.3.1 Impact of Moving Step Length

In general, when the moving step length D is small, the laser may have to point to many spots until triggering the camera, while a large D may lead to a long

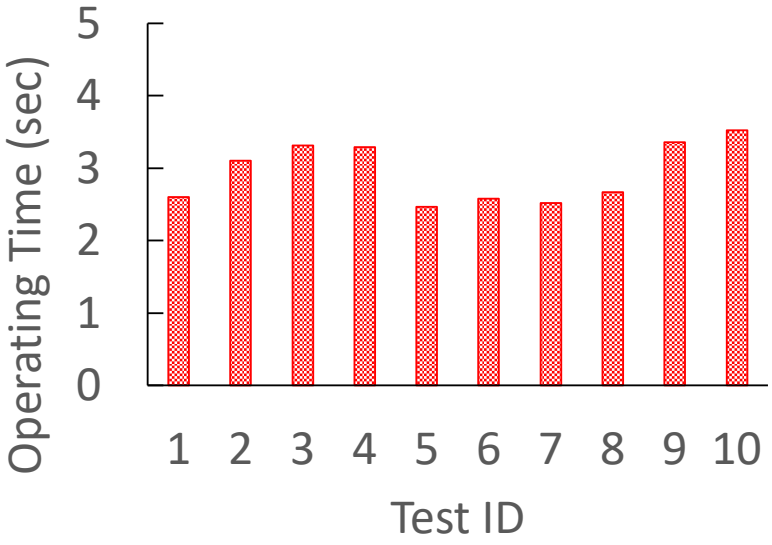


Figure 4.14: Operation time.

laser scan path. Accordingly, we change D from 0.1 to 1.0 m, with increments of 0.1. For each D, we perform 100 attempts of *PhantomMotion* to activate the camera. The camera's location and its motion detection range will be randomly changed after each attempt.

The success rate of *PhantomMotion* is 100% for all trials. Figure 4.18 illustrates the operation time when the moving step length varies. We observe that the median operation time is inversely proportional to D. Specifically, it is 1.5 sec when D is 1.0 m, and becomes 17.8 sec when D decreases to 0.1 m. This is because that a small D requires the laser beam to heat more points until the target device is triggered. Figure 4.19 presents the relationship between D and the scan distance. We see that the median scan distance increases with D. Particularly, when D is 0.1 m and 1 m, the corresponding median scan distances are 0.8 m and 3.2 m, respectively. These results demonstrate that selecting D is a tradeoff between scan distance and operation time. To achieve a desired scan distance

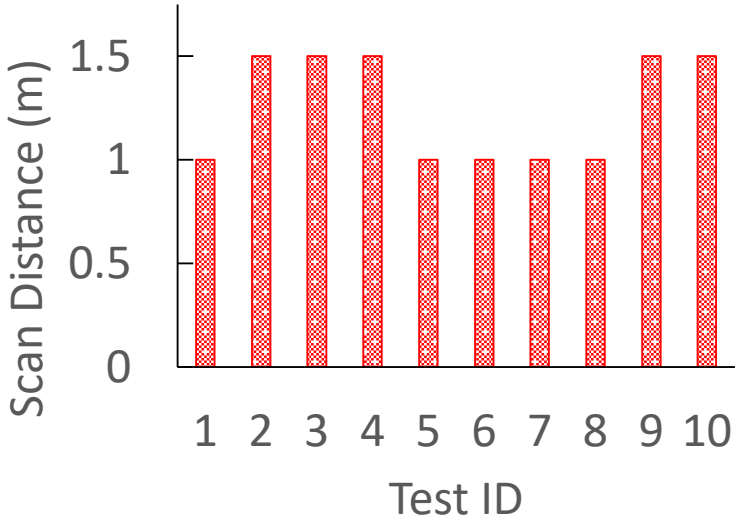


Figure 4.15: Scan distance.

(less than 1 m) and a comparably short operation time, we employ $D = 0.2$ m for the following discussions.

4.5.3.2 Impact of Heating Material

The motion detection area of the camera may have different surface materials, which may absorb varying laser energy, as discussed in Section 4.4.4.2. We consider four commonly used residential materials: brick, aluminum (abbreviated as alumi), stone, and wood. Their specific heat capacities are 800, 900, 1,000, and 1,700 J/kg $\cdot$ K, respectively.

For each material, we perform 100 trials of *PhantomMotion*.

Again, we achieve a 100% success rate of triggering the target device for all heating materials. Figure 4.20 presents the operating time for different materials. We can see that the operating time increases with the specific heat capacity of the material, while it remains consistently small (less than 10.5 sec) for differ-

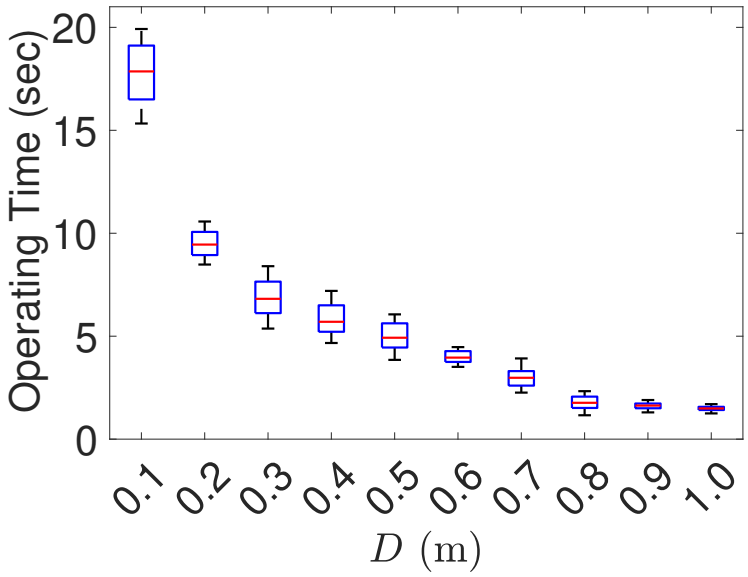


Figure 4.16: Time vs. D.

ent materials. Figure 4.21 shows the corresponding scan distances. We observe that the scan distance slightly changes with the heating material. The median scan distances for all materials are below 1.0 m. These results demonstrate that *PhantomMotion* is robust to the heating material.

Besides, *PhantomMotion* easily adapts to varying heating materials even in NLOS scenarios, as shown in Section 4.9.

4.5.3.3 Impact of Laser Transmission Path Length

The laser may be fired at different distances away from the target wireless system. Accordingly, we vary the distance L between the laser and target system (i.e., laser transmission distance) from 7 to 15 m, with increments of 1. For each L, we perform 100 attempts of *PhantomMotion* and all succeed.

Figure 4.24 shows the obtained operating time. We see that the median operation time is always less than 10 sec. Also, with L increasing, the operation

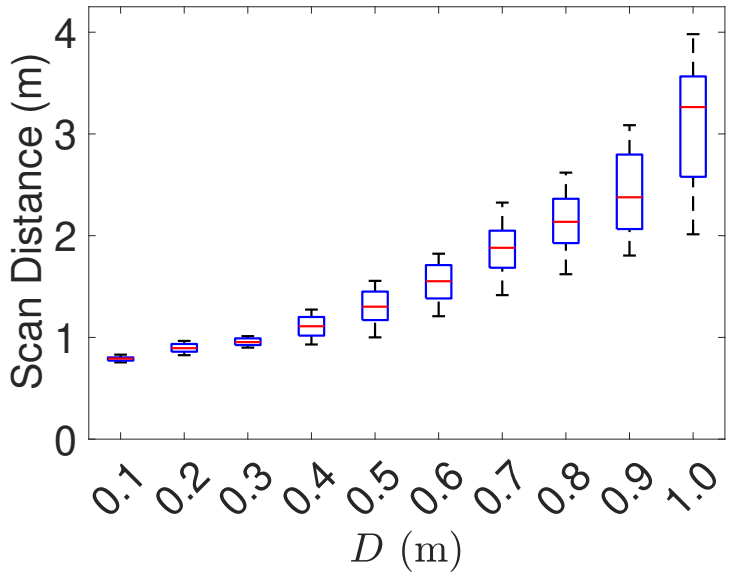


Figure 4.17: Distance vs. D.

time slightly decreases. This appears due to the fact that for a larger L, the required angle that the laser needs to turn is smaller for moving the laser point with a distance of moving step (i.e., D). The decreased time for laser angle adjustment lowers the operation time. Figure 4.25 presents the corresponding scan distances. We observe that for different L, the scan distance ranges are similar and the median scan distances are all less than 0.9 m, demonstrating that L has little influence on the scan distance. This is because that the scan distance is mainly affected by the moving step distance D, which remains constant.

Long-distance Tests: Empirically, it is difficult to focus the laser with the small lenses for a longer laser transmission distance (e.g., above 20 m). To address this limitation, we adopt a telephoto lens (e.g., Opteka 650-1300mm [138]) to focus the laser, and demonstrate *PhantomMotion*, which achieves laser transmission distances of up to 120 meters (the maximum safety distance restricted by the testing environment), as shown in Figure 4.26. Our experiment was con-

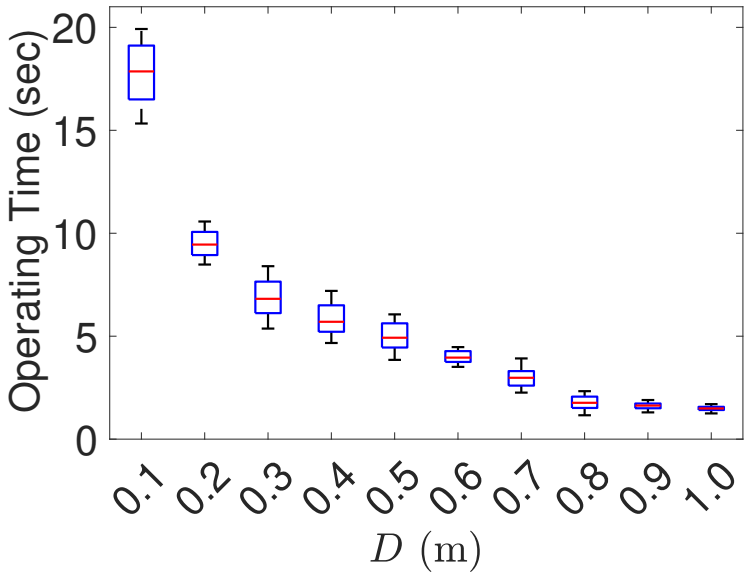


Figure 4.18: Time vs. D.

ducted within a secured 35-acre private farmland, enclosed by a fence, to ensure the utmost safety and avoid any potential risks. Lasers of class 3B, as used in our experiments, carry a potential risk of causing harm to the human eye and may even pose a fire hazard. To mitigate these risks, we restricted public access to our experimental area. Moreover, we carefully ensured that the laser was always directed towards the designated experimental area, further minimizing any potential fire danger.The target device connects to a popular commercial router - Netgear AC2600 (R7800) [134] in the house, which can cover up to 2500 square feet (i.e., 500 ft or 152 m away). We also use a scope (e.g., TRUGLO TRU-BRITE 30 TG8539TL [178]) for providing users with a clear view of targets up to 600 yards (about 549 meters).

We vary the laser transmission distance L from 20 to 120 m, with increments of 20, and perform 100 trials for each distance. We make sure that the environmental temperature maintains the same. We consistently obtain a 100% success

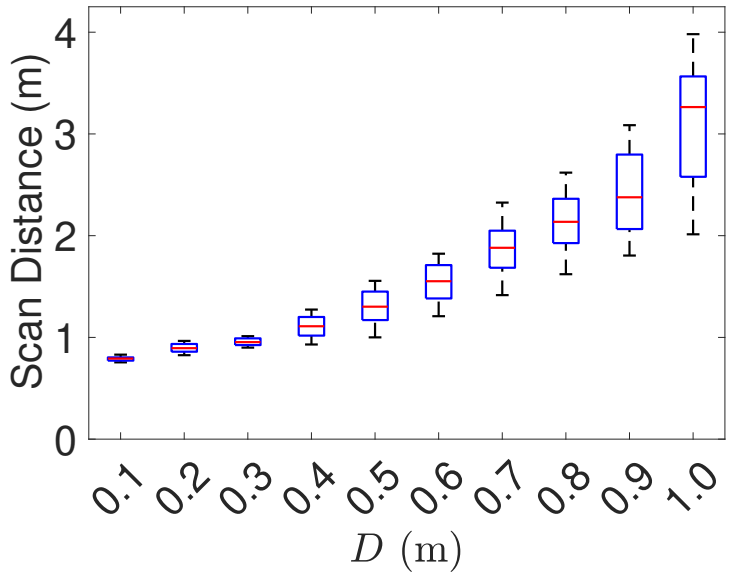


Figure 4.19: Distance vs. D.

rate. Figures 4.27 and 4.28 display the corresponding operation time and scan distances. We observe shorter operation time compared with short-distance tests. Specifically, the median operation time for 20 m is 8.9 sec, and it decreases to 8.2 sec when L equals 120 m. Again, a larger L requires a smaller rotation angle of the laser beam, leading to decreased operation time. Meanwhile, the average scan distance stays at 0.9 m regardless of L. The observation for scan distance is similar to that in short-distance tests.

Extended Range WiFi Traffic Capture: During our study, which involved using lasers to remotely activate motion sensors, we encountered skepticism regarding the feasibility of an adversary intercepting WiFi traffic from the distances under consideration.

In order to address these doubts, we designed an experiment involving 18 different security devices listed in Table 1. The tests were performed in two distinct settings: an apartment unit encumbered by multiple obstructions, such

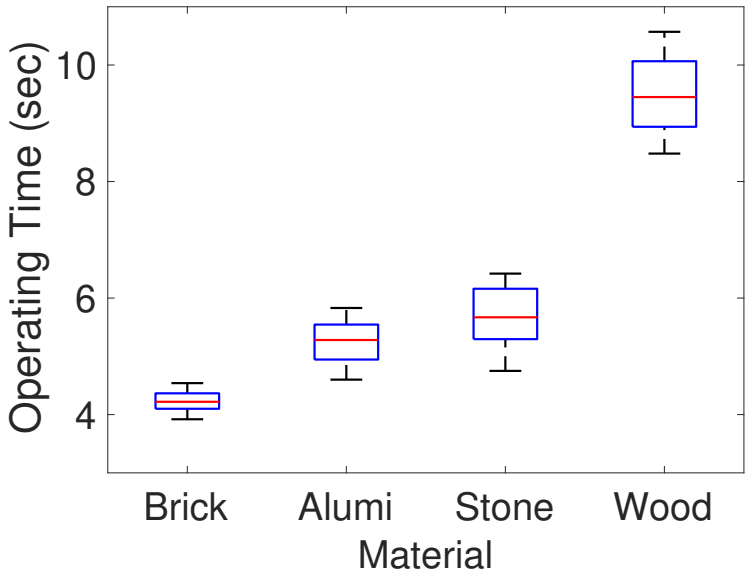


Figure 4.20: Time for different heating materials.

as walls, and a less obstructed environment in a farmhouse within an open field. For these evaluations, we used the same WiFi adapter for receiving signals, the Alfa AWUS1900, known for its claimed range of 500 feet in open areas [21].

Our protocol involved installing each security device one by one in both settings, and then continuously moving away from the device while keeping it in sight. We proceeded until no packets could be received from the security device, which allowed us to determine the maximum effective receiving distance. And the experiment results are presented in Figure 4.29, which illustrates the maximum receiving distances measured in both the apartment unit and the farmhouse for each device. The result revealed that the WiFi signals' reach was substantially hampered due to interference and physical obstacles. For instance, the device with the most potent signal, the Simplisafe Safety Alarm (Device 18), could only cover a maximum indoor distance of 37m, while all other devices had a range of 36m or less. The IHOXTX DF22 Cam (Device 8) had the least coverage, with a

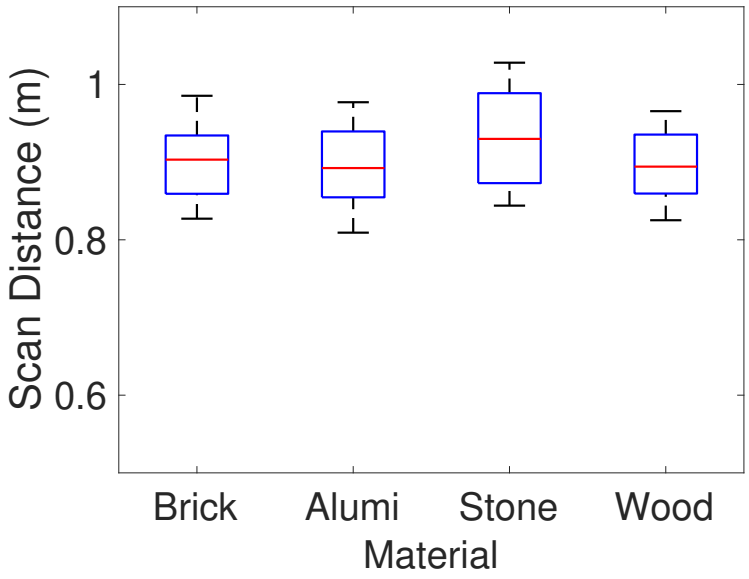


Figure 4.21: Distance for different heating materials.

maximum distance of 18m.

Conversely, when tested in the farmhouse, the devices displayed markedly improved wireless performance due to the absence of immediate physical barriers and reduced interference. We found that the signals from the security devices could cover considerable distances in open outdoor settings. Device 8, which had the smallest indoor range, managed to cover a maximum outdoor distance of 129m. Notably, Device 18 could reach as far as 167m. These findings empirically demonstrate that intercepting WiFi traffic from substantial distances, such as 120m, is indeed feasible given suitable conditions.

4.5.3.4 Impact of Environmental Temperature

We utilize a digital thermometer (Smart Thermostat Premium [53]) working with the HVAC to adjust the room temperature from 5 to 30 °C, with increments of 5. We conduct 100 attempts for each specified temperature.

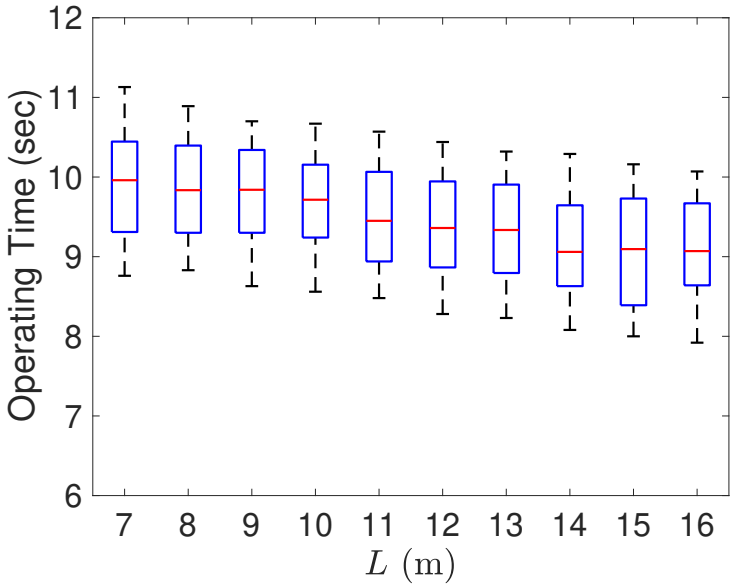


Figure 4.22: Time vs. L.

The success rate still maintains 100%. Figure 4.30 illustrates the operation time when the environmental temperature varies. We observe that the operation time slightly increases as the temperature decreases. Specifically, the median operation time at 30 °C is 5.3 sec, while it increases to 6.3 sec at 5 °C. This is because a lower room temperature would result in a longer laser heating time. On the other hand, the scan distance exhibits minor fluctuation with the temperature, as shown in Figure 4.31. Particularly, the median scan distance remains consistently below 0.9 m for varying temperatures. These results verify the robustness of *PhantomMotion* to the environmental temperature variation.

4.5.4 Overall Attack Impact

We develop a mobile app *PhantomMotion* and its designed user interface (UI) is presented in Figure 4.32, and we use the APP to perform the attack experiments. We perform 100 attack trials for each device and thus have $18 \times 100 = 1,800$

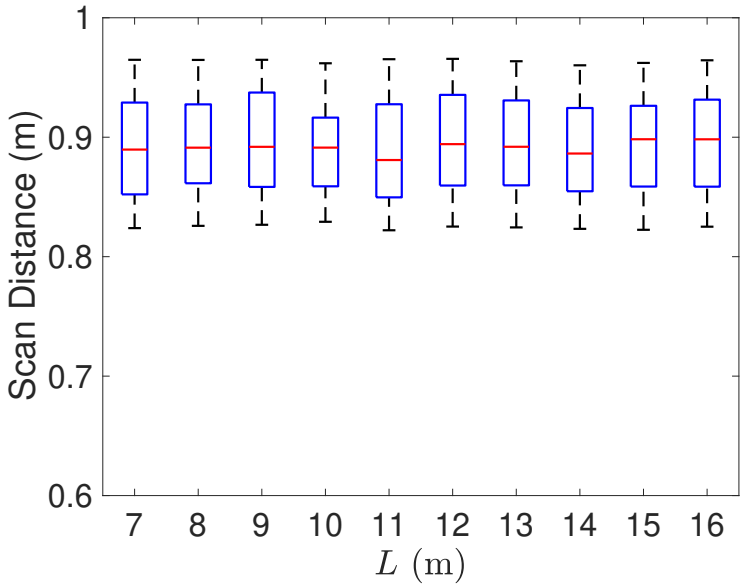


Figure 4.23: Distance vs. L.

attempts in total. Again, *PhantomMotion* achieves a 100% success rate. For each attempt, we record the corresponding operation time and scan distance, as shown in Figures 4.33 and 4.34.

We see three tendencies. First, *PhantomMotion* can always achieve a short operation time (ranging from 8.1 sec to 14.7 sec) and a small scan distance (from 0.8 m to 1.3 m). Second, on average, a device with two PIR sensors (Type 2, ID 11-13) causes a shorter operation time and requires a smaller scan distance than a device with one PIR sensor (Type 1, ID 1-10&14-18). This is because a Type 2 camera has a larger motion detection zone, causing the attacker to only need heating fewer positions for camera activation. Third, the performance is quite consistent across devices with the same amount of PIR sensors. For Type 1 or 2 devices, the mean operation times range from 13.2 sec to 13.8 sec, and 9.5 sec to 9.7 sec, respectively. Meanwhile, the respective mean scan distances stay around 1.2 m and 0.9 m.

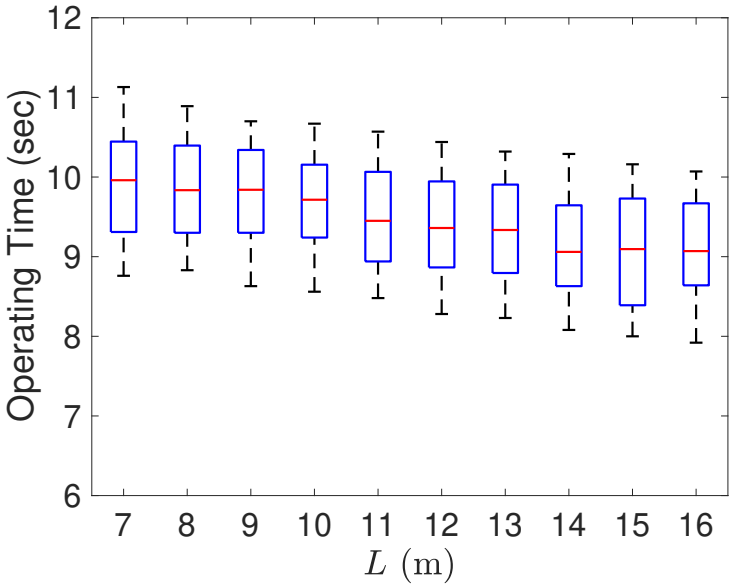


Figure 4.24: Time vs. L.

Figures 4.35 and 4.36 plot the CDFs of the operation time T_o and the scan distance S for different types of devices. We see that for Type 1, T_o is less than 14.5 sec with a 97.5% probability; for Type 2, T_o is less than 10.8 sec with a 98.3% probability. Also, S is less than 1.26 m for Type 1 with a probability of 92.0%, and it is less than 0.96 m for Type 2 with the same probability. These results again demonstrate that more PIR sensors may lead to a shorter operation time and a smaller scan distance, and also confirm conclusively that *PhamtonMotion* is robust against different devices.

In terms of potential false positives situations where the system could be triggered by other heat sources or radiation leading to inadvertent attack success, our experiments found none. Throughout our trials, we found that the camera's motion sensor was only activated when we intentionally moved the laser point into the detection area. We also accounted for other possible heat or radiation sources in our testing environment to ensure they did not affect our results.

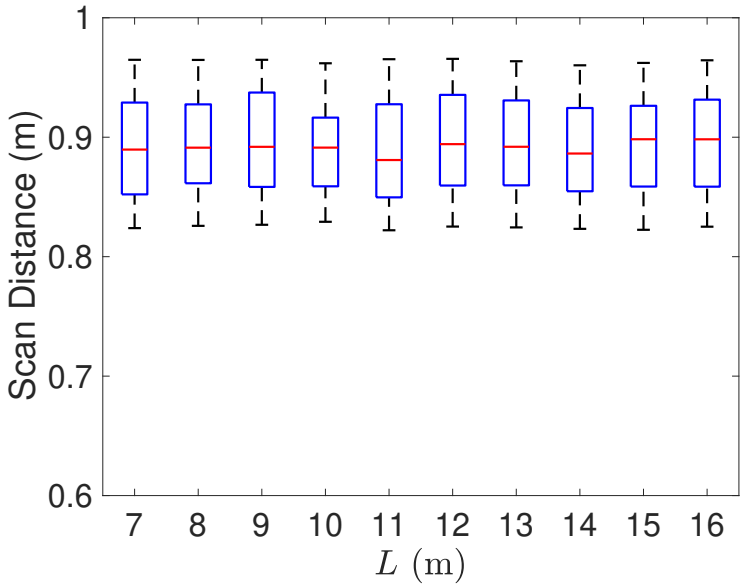


Figure 4.25: Distance vs. L.

4.5.5 User Study

We recruited 12 volunteers (U1-U12; aged 19-36 years old; 6 self-identified as female and 6 as males).[2] Before the experiments, all participants were trained on how to handle lasers safely based on our institutional guidance. Each participant was asked to perform *PhantomMotion* 100 times to trigger a wireless motion-activated device randomly selected and deployed at a random location inside the living room, as shown in Figure 4.12. For each deployment, we make sure that the device's motion detection area is not fully blocked.

Consistently, all users achieve a 100% success rate to trigger the target camera, confirming the attack feasibility.

Figure 4.37 shows their obtained operation time. We see that the maximum operation time for each user is always below 6.5 sec, and some users (e.g., users 3 and 7) can obtain an operation time of as short as 5.2 sec. Figure 4.38 presents the

[2]The study has been approved by our institution's IRB.

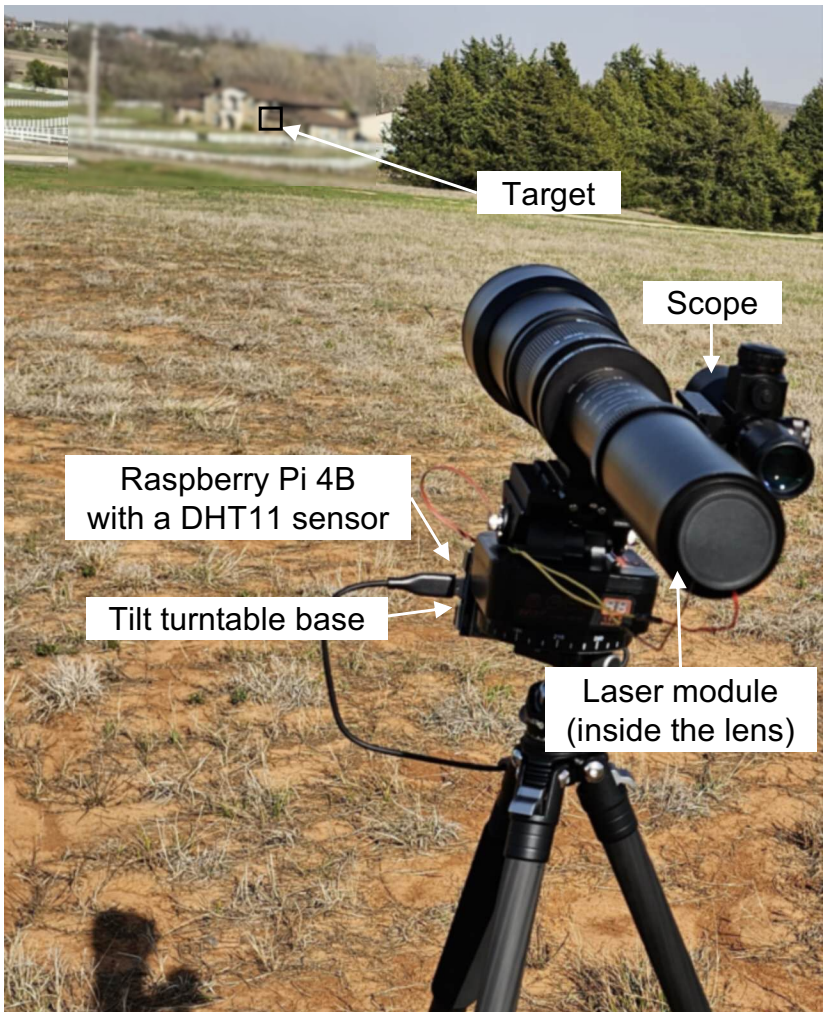


Figure 4.26: Laser aiming at the target across the 120 m field.

corresponding scan distances. We can observe a consistent median scan distance for all users varying between 0.88 and 0.91 m. These results demonstrate the attack efficiency is quite consistent among different users, and also verify the practicality of *PhantomMotion*. There is no false positive observed in user study experiments.

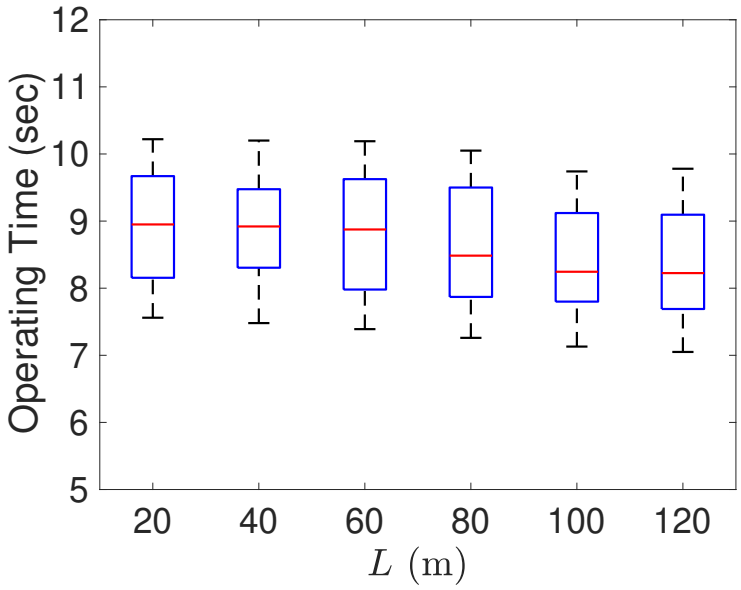


Figure 4.27: Operation time in long-distance tests.

4.6 Countermeasures and Limitations

4.6.1 Limitation

Line-of-sight Access: Same with all other light-based attacks, *PhantomMotion* also inherits the limitations of light-related physics, i.e., it requires line-of-sight (LOS) access to the target heating material. However, for *PhantomMotion*, the heating material does not necessarily lie in the LOS of the target wireless system due to radiation heat transfer. If obstacles have significantly low thermal conductivity rates or are too thick, the radiation may thus not be able to penetrate them, leading to the failure of *PhantomMotion*.

Customized Motion Detection Zones: Our experiments are performed with the security systems in factory default and recommended setting. Some systems do not support activity zone customization such as Simplisafe Cam and

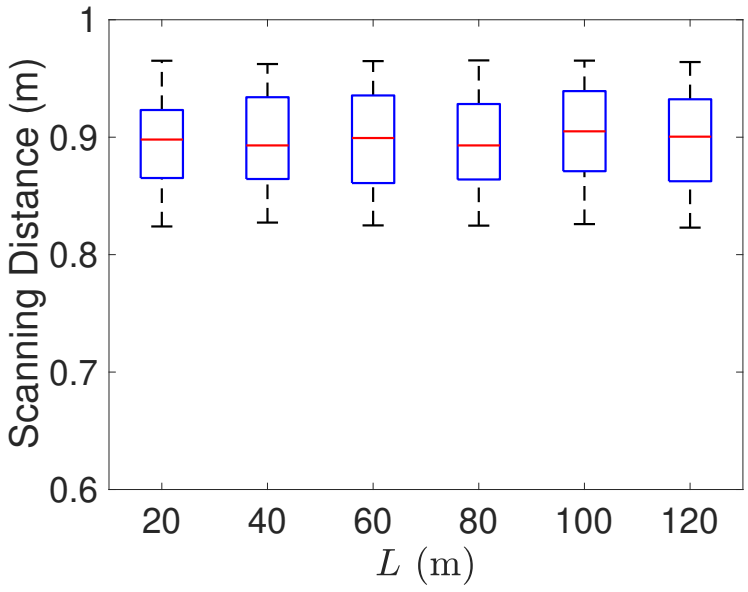


Figure 4.28: Scan distance in long-distance tests.

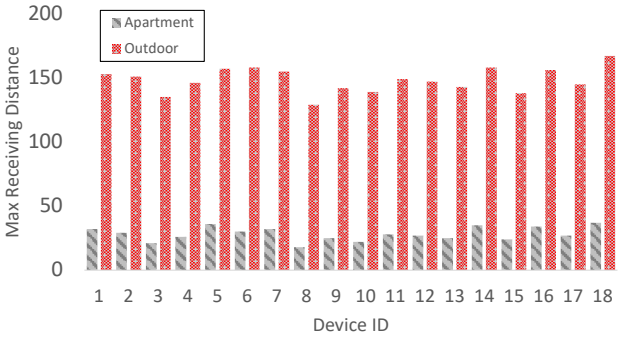


Figure 4.29: Max receiving distance for devices.

Ring Alarm System, while some allow users to manually create activity zones under certain circumstances (e.g., when the user purchases an extra subscription plan [33]). If a user manages to narrow the motion detection zone, *PhantomMotion* still works while it may need to scan a larger area, resulting in an increased operation time. Meanwhile, a decreased motion detection zone may also degrade the security system's ability of intrusion detection accordingly.

Security Systems Without WiFi Connectivity: The current implementation of *PhantomMotion* relies on sniffing IEEE 802.11 WiFi traffic to detect

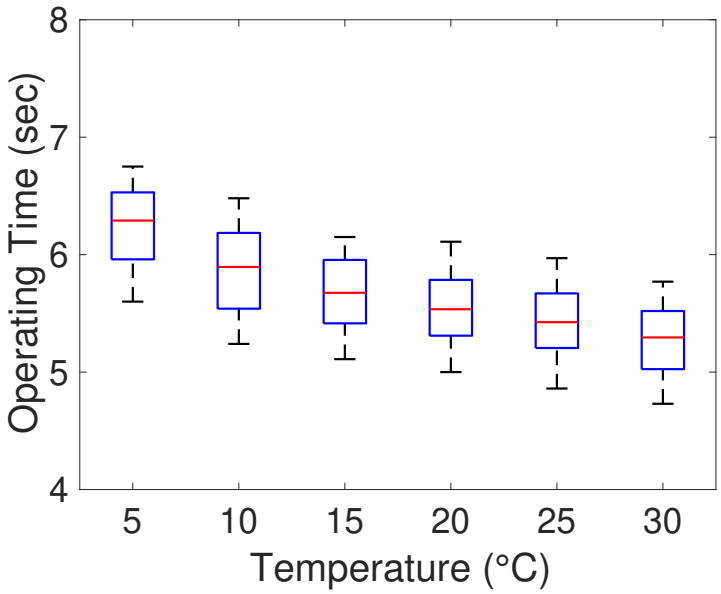


Figure 4.30: Operation time vs. environmental temperature.

the presence of target wireless systems and to infer whether they are activated. Consequently, it does not work against systems that communicate over non-WiFi networks, such as 4G LTE or 5G cellular networks operating on licensed spectrum. This limitation, however, can be addressed by equipping *PhantomMotion* with a Software Defined Radio (SDR), as demonstrated in prior work on cellular signal sniffing [36, 40], or with a commercial cellular analyzer such as Sanjole (acquired by Keysight Technologies) WaveJudge [99]. With such hardware, *PhantomMotion* can capture cellular control-plane signaling and extend its detection capability to systems that operate outside the WiFi spectrum.

4.6.2 Defense Strategies

PhantomMotion aims to trigger motion-activated wireless systems by leveraging laser heating to simulate human-generated radiation and wireless traffic sniffing for a response to that stimulation. Intuitively, to defend against *PhantomMo-*

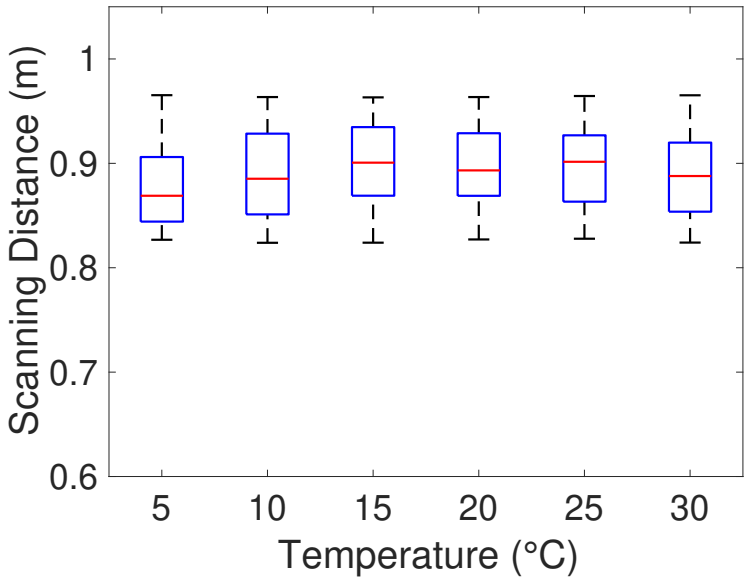


Figure 4.31: Scan distance vs. environmental temperature.

tion, we can stop the attacker from obtaining correct motion-correlated traffic. Accordingly, one straightforward defense is to *manually override and turn off the motion sensor* ([146]), causing the system to go into sleep mode and not respond to any motions (including spoofed ones). However, such a solution is impractical as the security system is no longer on the lookout for intrusion and will not timely send event alerts. Meanwhile, if the security system (e.g., a camera) is always active, the battery will be drained quickly and it may bring inconvenience to identify what you are looking for in the system's timeline.

A practical way to detect the attack is to incorporate another type of sensor (e.g., an ultrasonic detector [150], RGB color sensor [188], wireless transceiver [15, 125, 184, 196], or geophone [79, 128]) that is not affected by injected thermal radiation for verifying the physical motion. However, all those techniques require deploying extra hardware and effort such as synchronizing data obtained by the motion sensor and the other chosen sensor. Also, each new sensor has its own

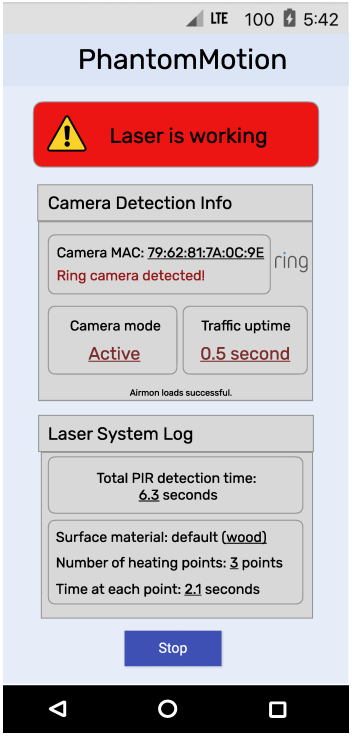


Figure 4.32: The UI snapshot of *PhantomMotion* when the target camera is activated.

limitations and may potentially introduce a new attack surface. For instance, an ultrasonic sensor may mistake air flows (e.g., air turbulence from HVAC systems) as human motion [76].

In a more expedient way, a wireless camera can analyze the recorded video after being triggered, verifying the authenticity of the detected motion. Nevertheless, this solution entails advanced computer vision, machine learning, or deep learning algorithms to distinguish human motion (e.g., [23, 80, 167]). It thus not only leads to delay of intrusion detection, but also requires the camera to equip with powerful video processing capability, causing increased energy consumption and device cost. Also, if the fake motion appears only in the non-overlapping area that belongs to the motion detection zone but not the field of view of the camera, this approach does not work as the camera is unable to capture the posi-

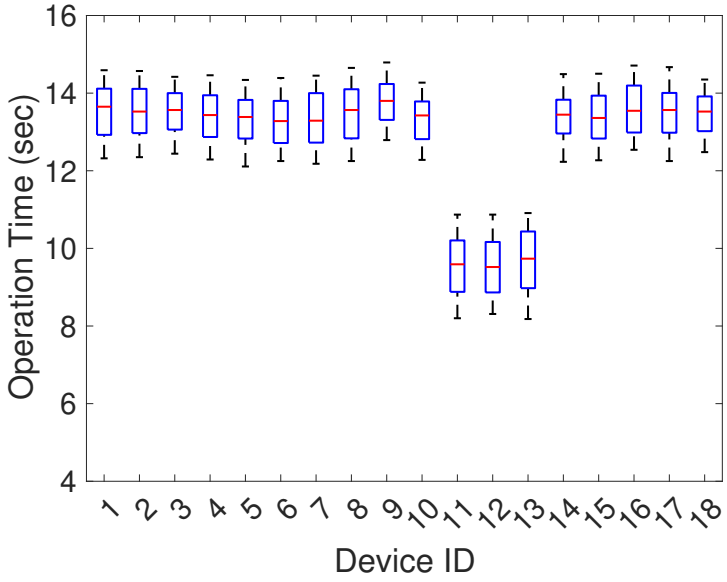


Figure 4.33: Overall operation time.

tion where the motion occurs. Besides, performing detection on raw surveillance videos may reveal the privacy of innocent people in the video while performing detection on encrypted videos needs additional efforts such as secret sharing [75]. Meanwhile, video-based systems may suffer from image injection attacks [116], where an attack can make a camera misperceive an actual scene or perceive a non-existent scene.

4.7 Related Work

Laser-based Attacks: Recent studies have managed to use lasers to attack different systems, such as unmanned aerial vehicles (UAVs) [49], voice-controllable systems [166], Light Imaging Detection and Ranging (LiDAR) systems [42, 141], and image recognition systems [191]. For example, by taking advantage of the observation that microphones often unintentionally respond to light as if it was

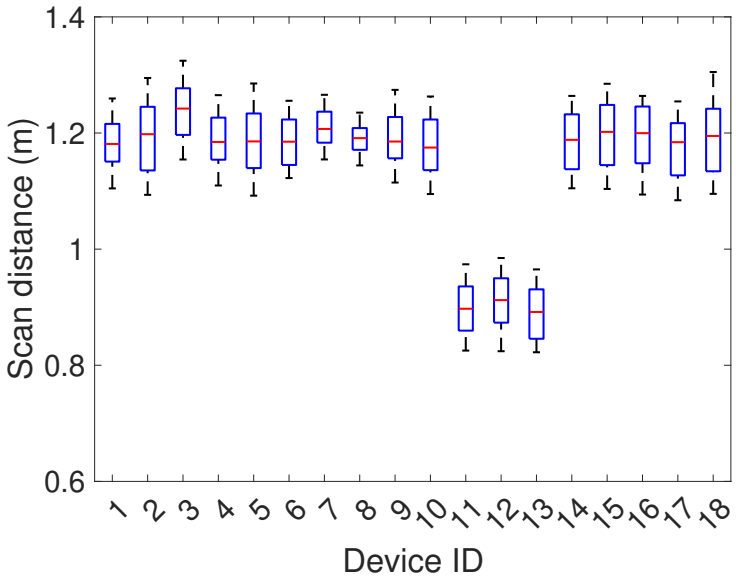


Figure 4.34: Overall scan distance.

sound, [166] can successfully inject commands into voice-controllable systems via laser light. Also, by exploiting the rolling shutter of the CMOS sensor, [191] manages to fool traffic light recognition with a laser. However, such attacks require the laser directly to point at a small area, such as one microphone port of the target device [166] or the camera on the target vehicle [191], and thus need to first pinpoint the target device/camera. In contrast, our work is the first laser-based attack targeting widely-deployed commercial wireless security systems. Also, it does not need to localize the target system in advance as the motion detection range is normally much wider and the proposed laser scanning method can help determine the laser destination.

Wireless Camera Detection: Existing research efforts to detect a wireless camera mainly include optical reflection based, thermal emission based, and wireless based.

The lens of a camera has to be exposed to monitor an area. When we shine

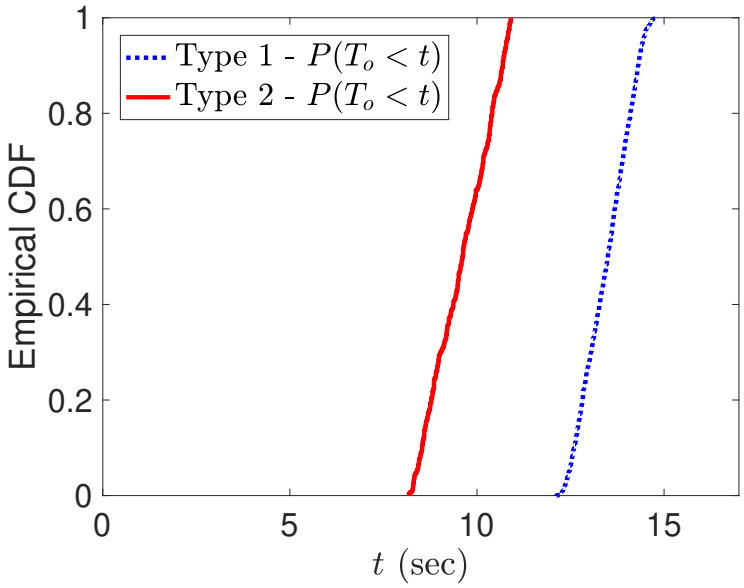


Figure 4.35: CDFs of operation time T_o.

a light, reflections may be observed as the light bounces off the lens. Traditional optical detectors, such as SpyGuy Camera Detector [165], emanate red light to assist users' subjective judgment with the reflections. A recent study [152] proposes to detect cameras by identifying unique high-intensity reflection from camera lenses. However, such optical reflection based methods require the user to be in close proximity (e.g., 0.5 m) to the target camera and scan every corner of the target area. The near-distance and frequent scanning process may inevitably disclose the user to the camera.

Also, thermal cameras can reveal heat traces on devices. The work [195] develops a Deep Neural Network (DNN) model to detect cameras by leveraging their heat dissipation patterns in thermal camera images. Nevertheless, such a thermal emission based method needs a training process to pre-build a data set consisting of visual and thermal images, which may not be possible in a strange or uncontrollable environment. If a camera model is unseen in training, the

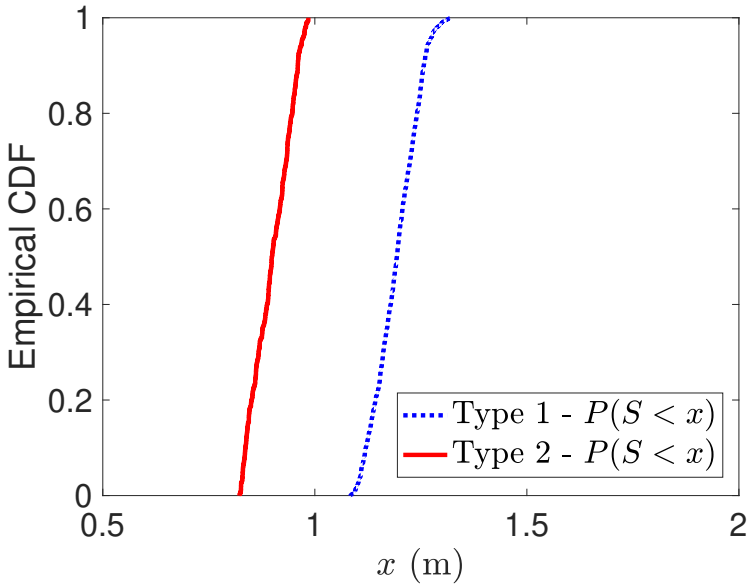


Figure 4.36: CDFs of scan distance S.

detection performance degrades accordingly. Meanwhile, this technique needs the user to repeat trials of taking pictures in front of the target camera, causing a great risk of disclosing the user.

Recently, many studies have shown the success of leveraging wireless traffic to detect wireless cameras (e.g., [44, 45, 86, 91, 151, 160, 162]). By stimulating the target camera with human motion, the resultant wireless traffic may disclose the existence or even the exact location of the camera. However, all these techniques require the user to laboriously perform motion in front of the camera, and most of them even require the user to carry a smartphone to sense the motion via an accelerometer. Such preconditions may cause significant discomfort to the user, and also bring risks of getting caught. *PhantomMotion* gets rid of real human motion and can trigger wireless cameras remotely. We compare our work and previous wireless camera detection schemes in Table 4.2.

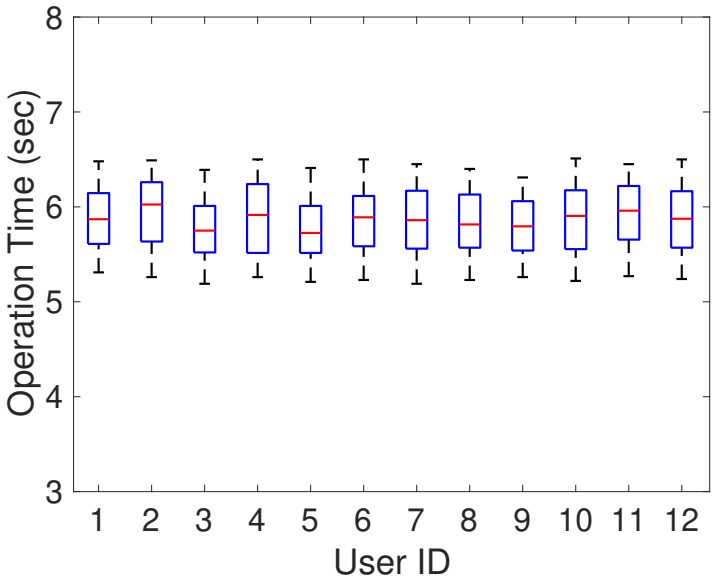


Figure 4.37: Users' operation time.

4.8 Conclusion

We present *PhantomMotion*, a novel technique for remotely triggering wireless motion-activated security systems (such as widely deployed IoT cameras and alarm systems) with a laser. It is the first to activate security systems requiring neither penetrating into the same network with target systems nor physical human motion in close proximity to the systems. By exploiting the working mechanisms of motion sensors, we manage to create fake motion signals and inject them into an area to stimulate a security system monitoring this area to emit wireless traffic, which can be collected and correlated with the laser-based stimulus to check whether the target system activates. Our real-world evaluation with 18 popular off-the-shelf wireless security systems demonstrates the effectiveness and efficiency of *PhantomMotion* under varying conditions.

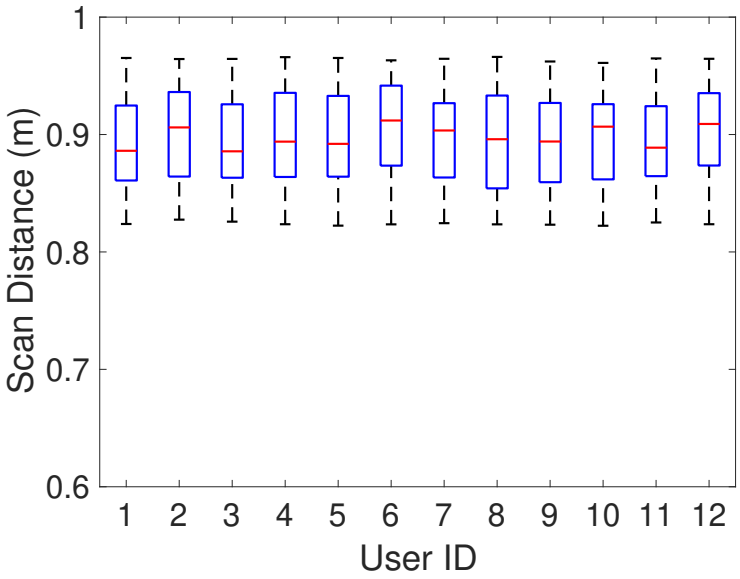


Figure 4.38: Users' scan distances.

4.9 Non-Line-of-Sight Scenarios

We test a wall with six different materials that can block visible lasers, including aluminum (alumi for short), brass, copper, brick, stone (lime), and wood. Such metal and non-metal materials are widely used for building walls or various everyday objects. A fraction of a side of the wall is inside the motion detection range of the camera, while we then shoot the laser beams from the opposite side of the wall. As a result, a fraction of the heat would traverse the wall, and may then trigger the camera. Meanwhile, the camera is unable to record the laser beams when it is activated.

Table 4.3 presents the thermal conductivity rates (referred to as λ, denoting the intrinsic ability of a material to conduct heat [77]) of tested materials. They are measured in watts per meter kelvin (W/mK) and divided into two categories (high and low). Empirically, as the thermal conductivity rate of wood is too low,

Table 4.2: Comparison with prior research efforts in wireless camera detection.

System	Motion in Front of Camera	Carrying a Device	Motion Type	Motion Duration
DeWiCam [44, 45]	✓	✓ (phone)	Walking/halting, waving hands, jumping	Default: 15 sec
MotionCompass [86]	✓	✓ (phone)	Walking along a specific route	Avg: 135–143 sec
SCamF [91]	✓	✓ (phone)	Standing still; waving hands in varying postures	30 sec (20 standing + 10 moving)
SnoopDog [162]	✓	✓ (phone & laptop)	Stop-start-stop; jumping jacks; walking; standing still	40 sec (detect.); $30n$ sec (n-trial loc.)
Lumos [160]	✓	✓ (phone/tablet)	Walking around the space's perimeter	Within 30 min
CSI:DeSpy [151]	✓	×	Daily activities (e.g., changing dresses)	8 sec or longer
PhantomMotion	×	×	None	0

Table 4.3: Thermal conductivity rate λ (W/mK).

	High (metal)			Low (non-metal)		
Material	Copper	Brass	Alumi	Stone	Brick	Wood
λ	398	146.87	130	1.3	1.03	0.13

we find that the corresponding heat transmission is too low-efficient to generate phantom motion signals within an appropriate time (e.g., several minutes). We then focus on obstacles with the rest five materials. For each material, we perform 100 trials of *PhantomMotion*.

The success rates for all five materials are 100%, convincingly indicating that the proposed attack can be launched in NLOS scenarios. Table 4.4 presents the mean, minimum, and maximum operation time for different materials. We observe the operation time is almost inversely proportional to the material's thermal conductivity rate, which indicates the thermal transfer efficiency. A high thermal conductivity rate would reduce the laser heating time at each point along the scan path. Specifically, for all metal materials, the average operation time

Table 4.4: Operation time vs. material in NLOS scenarios.

Material	Operation Time (seconds)		
	Average	Minimum	Maximum
Copper	5.7	5.3	6.2
Brass	6.7	6.2	7.1
Aluminum	10	9.7	10.3
Stone	308.4	299.4	319
Brick	399.3	392.6	406.5

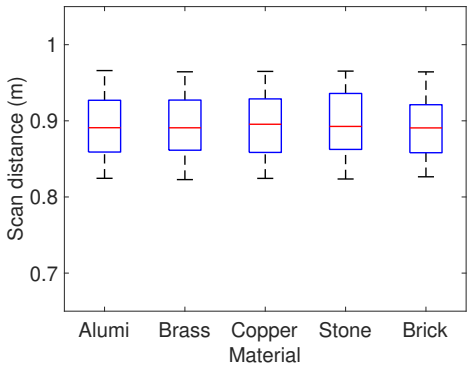
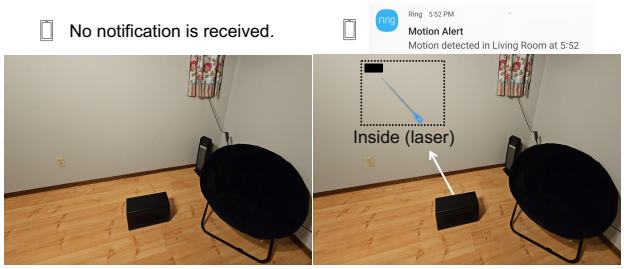


Figure 4.39: Scan distance vs. material.

is constantly within 10 seconds; for non-metal materials - Stone and Brick, their operation times jump to around 5∼7 minutes. Figure 4.39 plots the corresponding scan distances. We observe that the scan distances for different materials are similar, within the range of 0.82 m to 0.97 m. The material thus has no apparent impact on scan distance. These results demonstrate that *PhantomMotion* can be robust to different heating materials even in NLOS environments.

Examining the Role of Material Thickness: In our preliminary Non-Line-of-Sight (NLOS) experiments, we consistently used materials with a thickness of 1 cm. Given that the thickness of a material can significantly influence heat transfer and dissipation, it was deemed crucial to evaluate this factor's potential effects on our NLOS tests. We chose aluminum as our subject material for these investigations. Acquiring aluminum sheets in varied thicknesses presented a challenge, so we sourced 6061 T651 Aluminum Sheets of 1/16, 1/8, 1/4, 1/2, and



(a) No motion is detected with no attack. (b) Motion is detected with the attack.

Figure 4.40: Camera feeds before and after the black-box attack.

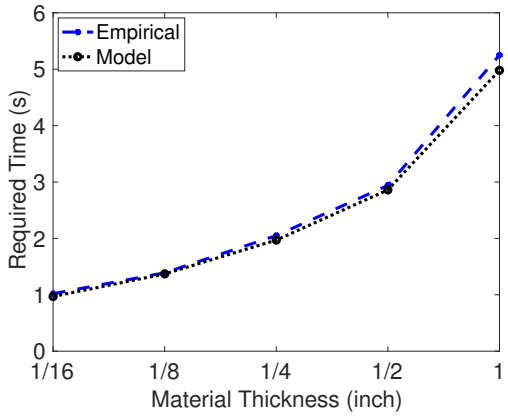


Figure 4.41: Operating time for different aluminum thickness.

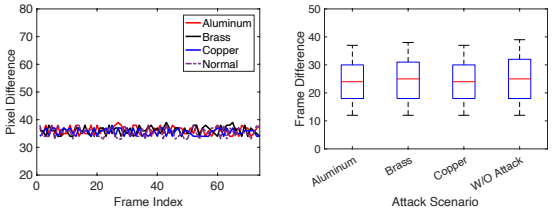
1 inch thicknesses from Amazon, ensuring that all sheets were identical in length and width. We monitored the temperature on the reverse side of each sheet until it reached the target temperature, T_t, recording the time taken for each thickness. We perform 50 experiment trials per thickness level and calculate the mean required time for each of the thickness. To establish a benchmark for the actual time required for the rear of the material to attain the target temperature, we also computed the predicted total time necessary for heat transfer to the back of the material. This prediction utilized Equations 5 and 6, with the total time calculated as the sum of t_{h_0}, t_{h_1}, and t_t. For these calculations, we considered the room temperature as 20 degrees Celsius and set the target temperature to 37 degrees Celsius.

Figure 4.41 illustrates the time taken for each thickness to reach T_t alongside

calculated predicted times. Notably, the observed and predicted times for laser heating were closely aligned, and both displayed an inverse relationship with the increase in thickness.

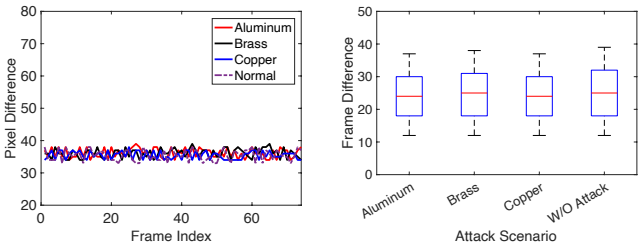
Special Case: "Black-box" Attack: The above analysis implies that the proposed attack is able to create fake motion to trigger wireless security cameras behind the walls, without leaving any evidence recorded by the target cameras. For a more covert attack, we develop a "black-box" attack method, in which we put a laser system in a closed box made of opaque materials that can transfer heat within a reasonable time, such as the five materials we test above. The laser can be controlled wirelessly via the smartphone. It can then heat the interior surface of the box, and also the corresponding exterior surface of the box via a spontaneous heat transfer process. When this exterior surface of the box happens to face the target camera, the generated fake motion signals may trigger the camera accordingly. Figure 4.40 demonstrates such an attack case, where *PhantomMotion* successfully triggers the camera. By comparing the camera feeds before and after the attack occurs, we see that there is no visual difference.

Frame-by-Frame Examination: An intriguing question arises regarding whether the laser-induced heating on a material's surface can cause detectable changes on the rear side of the material, potentially captured by the camera. To address this concern, we conducted experiments on three different metal materials, namely Aluminum, Brass, and Copper. For these experiments, we utilized the Ring Stickup Cam to capture video footage of the 5-second laser heating process for each material. Each 5 seconds video clip is 15 FPS (Frames Per Second) as pre-formatted by Ring [147], we collecting 100 clips per material so there is 15*5*100 = 7500 frames for each of the material. Consequently, we obtained 7500



(a) Frame Pixel Difference (b) Frame Difference CDF

Figure 4.42: Frame by Frame difference output.



(a) Pixel Difference in One Video (b) Frame Difference by Scenario

Figure 4.43: Frame comparison result.

frames per material to compare against a standard video recording (without laser heating) for similarity analysis.

To evaluate the similarity between each frame, we utilized a straightforward frame difference method, widely recognized for its effectiveness in detecting motion by comparing pixel differences between frames [101, 127, 164]. For example, the difference of the current frame and next frame is calculated as $I_{d(N,N+1)} = |I_{N+1} - I_N|$, where N refers to the Nth frame in the video clip, N + 1 is the succeeding frame, and $I_{d(N,N+1)}$ signifies the difference between these two frames.

We applied this method to compare two frames extracted from a video of an aluminium material being heated by a laser for a duration of 5 seconds. We randomly select one frame before heating and one frame after heating, then visualize difference.The pixel differences were visualized in Figure 4.42, showing only a few residual RGB noise, which indicates that the differences are merely random noise deviations.

To reinforce our conclusions, we first computed the pixel differences from a single video clip chosen from the 100 samples for each material, comparing these with a control case where no heating was applied. Figure 4.43 (a) demonstrates that during the heating process, all three materials—aluminium, brass, and copper—show a similar mean pixel difference of less than 40, mirroring the stability observed in a stationary video under normal conditions. In contrast, a typical video without heating exhibits a consistent difference of less than 40, indicating significant frame-to-frame consistency.

Figure 4.43 (b) illustrates the frame difference across four attack scenarios. The first three scenarios involve heating of three different materials: aluminium, brass, and copper. The last scenario is the normal case with no heating on any material. The frame differences for these scenarios, derived from 7500 frames from 100 video clips, fall within the range of 12 to 38. This closely mirrors the range of a non-heated video, which is from 12 to 39. Collectively, these findings corroborate that the subtle changes induced by laser heating remain undetectable by the camera being investigated.

4.10 Laser Classes

Different from other sources of light, a laser generates coherent and intense light. The American National Standards Institute's Z136.1 standard classifies lasers according to their relative hazard, including

- *Class 3a Lasers:* This class contains devices with a relatively low output power in the range of 1 to 5 mW. Such lasers are considered safe for brief eye exposure.

- *Class 3b Lasers:* This class has intermediate power (5-500 mW), and will not cause thermal skin burns or fires. The lasers may pose a moderate risk of eye injury.

- *Class 4 Lasers:* This class has high power (above 500 mW), and may cause an eye injury if the beam enters the eye. All eye exposure tob the beams should be avoided.

Chapter 5

Future Work

5.1 Embedded Systems and IoT Security

This dissertation has investigated the layered security vulnerabilities embedded within modern wireless security cameras, approaching the problem from two fundamentally different angles. Through *WeakCamID*, we demonstrated that adversaries can passively and reliably infer the operational state and subscription tier of wireless cameras purely by observing encrypted network traffic patterns, with no need to break any cryptographic primitive or touch a single packet's payload. Through *PhantomMotion*, we revealed that the physical sensing layer of these devices is equally fragile: simple, consumer-grade lasers can be weaponized to inject phantom motion signals into passive infrared (PIR) sensors, triggering recording events and alarm states across walls and structural barriers.

Together, these findings expose a foundational tension at the heart of the IoT security model, where the interplay between network-layer metadata and physical-layer sensor design creates attack surfaces that neither the cryptogra-

phy nor the hardware communities have fully addressed. Looking forward, the insights gained from these two research thrusts open a rich set of future directions extending beyond wireless cameras to a broader range of IoT systems, including Automatic Speech Recognition (ASR) devices [197]. The following sections describe six promising areas where future research can build meaningfully upon the foundations laid in this dissertation.

5.1.1 Defending Against Network-based Side-Channel Attacks via Adaptive Traffic Obfuscation

WeakCamID established that the volume, timing, and burstiness of encrypted IoT traffic accurately betray sensitive operational states, such as whether a camera is actively recording, sitting idle, or streaming a live feed to its owner. While the root cause is the subscription-tier differential behavioral model imposed by manufacturers, the broader problem is endemic to virtually all IoT ecosystems, where traffic patterns are dictated by physical events and thus inherently non-random. A more robust and universally applicable remedy lies in intelligent traffic shaping at the device level.

Future work must focus on designing efficient obfuscation frameworks specifically engineered for resource-constrained IoT hardware [27]. Conventional mitigation strategies, such as continuous dummy traffic injection or constant-rate padding, are deeply impractical for battery-powered smart cameras. These approaches impose continuous energy overhead that can slash battery life from months to days, which is simply unacceptable for consumer products. Instead, future research should investigate demand-aware, adaptive obfuscation schemes that activate only when a physical event is detected, selectively masking the

outgoing traffic burst without flooding the network during idle periods. One promising avenue is deploying lightweight generative models directly on the camera's system-on-chip (SoC). Such a model would learn the statistical footprint of legitimate traffic during normal operation and generate plausible cover traffic that statistically mimics active states, regardless of the true camera state. The key challenge here is maintaining the fidelity of this cover traffic across diverse camera models and network conditions, because a one-size-fits-all approach will inevitably create its own detectable fingerprint.

5.1.2 Robust Multimodal Sensor Fusion for Surveillance Security Architectures

The consistently high attack success rate achieved by *PhantomMotion* ultimately stems from a single design philosophy that currently dominates the consumer camera market: entrusting the entire wakeup decision to a single, unauthenticated analog sensor. PIR sensors are cheap, low-power, and effective under normal operating conditions, but they are also completely blind to the source of the thermal stimulus they detect. An adversary with a handheld infrared laser can trigger the same detection response as a person walking through a room.

Future IoT security architectures must migrate toward robust multimodal sensor fusion, where a spurious or anomalous reading from one sensor cannot unilaterally drive a high-stakes state transition. Methodologies inspired by the Sensor Defense In-Software (SDI) paradigm [175] offer a useful starting point: machine learning models trained to detect contradictions between simultaneous sensor readings can flag physically impossible event combinations. For example, a PIR trigger that is not accompanied by any corresponding acoustic event,

vibration, or millimeter-wave (mmWave) radar return is deeply suspicious. Incorporating supplementary, low-power sensing modalities such as UWB ranging or even passive acoustic monitoring can provide dimensional cross-validation before the camera commits to its recording state. The research challenge lies in implementing these fusion algorithms within the severe memory, computation, and energy budgets of embedded camera hardware, while ensuring that the fusion logic itself does not become a new attack surface for adversarial sensor manipulation.

5.1.3 Studying Attack Propagation in Smart Home Trigger-Action Ecosystems

Neither *WeakCamID* nor *PhantomMotion* was designed with the broader smart home ecosystem in mind, yet both attacks have implications that reach far beyond the camera itself. Modern smart home deployments are increasingly built around Trigger-Action Programming (TAP) platforms such as SmartThings, Apple HomeKit, Google Home, and IFTTT, where physical events detected by one device can autonomously actuate another. A camera detecting motion might unlock a front door, illuminate exterior lights, or notify a neighbor's security system. This interconnectedness is a feature, but it also means that a single spoofed physical event can cascade through the home's automation logic in ways the owner never anticipated.

Prior research by Wang et al. has systematically shown that the overlapping interactions among legitimate automation rules in smart homes can create severe, exploitable logical vulnerabilities [185]. Future research should formalize the compounding risk that physical spoofing attacks like *PhantomMotion*

pose within these trigger-action architectures. Concretely, an attacker who uses laser injection to generate a fake motion trigger could potentially unlock a smart door lock, disable a smoke detector, or suppress a perimeter alarm, all without ever connecting to the home network. Addressing this threat requires decentralized, formally verified event validation mechanisms that cross-check sensory event claims against multi-source physical evidence before allowing them to propagate commands to high-integrity actuators.

5.2 Security Threats in Large Language Model Training Pipelines

The rapid proliferation of large language models (LLMs) as foundational infrastructure for information retrieval, code generation, and decision support has introduced a new class of integrity attacks that share a deep conceptual kinship with the sensor spoofing explored in this dissertation. Just as *PhantomMotion* injects a deceptive physical signal into a sensor's input pipeline to manipulate its output, adversaries can inject deceptive data into an LLM's training pipeline to manipulate its learned behavior in ways that remain dormant until a specific trigger is encountered.

The seminal work of Gu et al. on BadNets [73] formalized the concept of backdoor attacks in deep neural networks, establishing that models trained on poisoned datasets can achieve normal performance on clean inputs while producing attacker-specified outputs whenever a pre-agreed trigger pattern appears. More recently, Wan et al. extended this threat to the instruction-tuning phase of modern LLMs, demonstrating that poisoning even a small fraction of fine-tuning

examples can persistently corrupt the model's behavior on downstream tasks in a targeted and stealthy fashion [183]. Goldblum et al. surveyed the full landscape of dataset security for machine learning, underscoring that data poisoning and backdoor attacks represent a systematic and largely unsolved problem across the entire machine learning pipeline [69]. In federated learning settings, Bagdasaryan et al. showed that a single malicious participant can execute a model-replacement attack that implants a backdoor into the global model with dramatically higher success rates than simple data poisoning [37].

Future research should examine how the physical and network side-channel methodologies developed in this dissertation can be adapted to detect, characterize, and ultimately prevent training-time integrity attacks on LLMs. For instance, the behavioral fingerprinting techniques underlying *WeakCamID* could inspire output traffic analysis approaches to detect anomalous LLM response distributions that betray the presence of a latent backdoor trigger. More broadly, building upon recent advances in human input inference [96, 193], future work should investigate whether the keystrokes, speech patterns, or interaction behaviors of users interacting with a backdoored LLM leave a detectable physical or network footprint that could serve as an out-of-band early warning signal.

5.3 Electromagnetic Emanation as a Pervasive IoT Privacy Side-Channel

The traffic side-channel exploited by *WeakCamID* treats the wireless network as the eavesdropping medium. However, every digital circuit that processes or transmits data simultaneously radiates electromagnetic (EM) energy as an unavoidable

byproduct of its operation. This unintentional radiation, known since the foundational work of Kocher et al. on differential power analysis [103] and further explored by Mangard et al. in the context of embedded cryptographic implementations [117], constitutes a physical side-channel that can leak secret information without any network access whatsoever. Vuagnoux and Pasini demonstrated this threat concretely in the context of keyboards, recovering typed keystrokes from electromagnetic emanations at distances of up to 20 meters through walls [182].

For the IoT ecosystem, EM leakage represents a largely underexplored threat. Modern wireless cameras, smart locks, and home automation hubs all contain processor cores, memory buses, and radio transceivers that radiate as they execute code, process sensor data, and manage cryptographic operations. A passive EM eavesdropper positioned in the vicinity of a home could potentially extract sensitive operational state information from these emanations without triggering any network-based intrusion detection mechanism. Future research should investigate the extent to which EM emanations from IoT edge devices leak inferrable information about their operational states, user interactions, and cryptographic key material. This direction is especially compelling in scenarios where network access is impossible, such as air-gapped deployments or environments with strict traffic monitoring, yet physical proximity to the device is achievable. Building upon the physical measurement methodologies developed for *PhantomMotion*, a promising direction for future work is to adapt near-field EM capture techniques to characterize the radiation profiles of commodity IoT devices and to develop the signal processing pipelines needed to extract meaningful state information from raw EM traces.

5.4 Security of Programmable Logic Controllers in Infrastructure

The vulnerabilities uncovered in this dissertation ultimately concern the boundary between physical phenomena and digital decision-making. This same boundary, when instantiated in industrial environments rather than consumer smart homes, becomes the foundation of an entirely different and far more consequential threat landscape: the security of Programmable Logic Controllers (PLCs) in critical infrastructure. The Stuxnet campaign, thoroughly analyzed by Langner [107], demonstrated with devastating clarity that cyber attacks targeting PLC logic can cause real physical destruction, silently pushing centrifuges beyond their operational limits while reporting nominal behavior to operators. McLaughlin et al. provided a systematic survey of the cybersecurity landscape governing industrial control systems, documenting how legacy protocols, flat network architectures, and the assumption of physical isolation have left PLCs exposed to a wide range of attacks that are increasingly viable given the ongoing convergence of IT and OT networks [123]. McLaughlin et al. further showed that even the trusted safety logic running on PLCs cannot be assumed benign without formal verification, proposing a trusted safety verifier that examines PLC code before deployment [124].

Future research should investigate how the sensing and injection techniques developed in this dissertation apply to PLC environments. Specifically, the laser-based thermal injection methodology from *PhantomMotion* could, in principle, be redirected at the temperature and vibration sensors that PLCs use to monitor industrial equipment health. A PLC that receives a spoofed nominal temperature

reading from a tampered thermal sensor might suppress a protective shutdown, allowing real physical damage to accumulate undetected. Similarly, the traffic fingerprinting methodology from *WeakCamID*, applied to the Modbus, EtherNet/IP, or Profibus communications between PLCs and their supervisory control systems, could reveal sensitive production schedules, safety setpoints, and operational configurations to a passive network adversary. Together, these extensions would open a new and practically significant research agenda at the intersection of physical-layer spoofing, network-layer metadata leakage, and industrial control system security.

Concluding Remarks The vulnerabilities documented in this dissertation are not isolated bugs in particular camera models. They are symptoms of a deeper mismatch between the security assumptions baked into modern IoT design and the adversarial realities of real deployment environments. Encryption protects data payloads but not behavioral metadata. Analog sensors detect physical stimuli but cannot verify their authenticity. Automation rules compose individual safe behaviors into collectively unsafe outcomes. These tensions will only intensify as smart devices multiply, as LLMs become embedded in decision-critical pipelines, and as operational technology systems converge with general-purpose IT infrastructure. Addressing them fully will require an intentional interdisciplinary effort that draws on signal processing, hardware security, machine learning theory, and formal verification in equal measure. The work presented here is a step in that direction, and the future research agenda outlined above aims to keep the security community ahead of an adversarial landscape that is evolving at the same pace as the technology it targets.

Chapter 6

Conclusion

This dissertation has systematically investigated the pervasive security and privacy vulnerabilities embedded within modern wireless surveillance systems. Despite the widespread deployment of payload encryption and the physical isolation of smart home environments, we demonstrated that commercial security cameras remain remarkably susceptible to both network-layer side-channel inferences and physical-layer sensor spoofing.

In our first work, we introduced *WeakCamID*, a novel, passive traffic analysis attack that exploits the distinct operational states introduced by cloud-based subscription models. We revealed that the structural differences between subscription-tied motion recording and local streaming manifest as highly identifiable patterns in encrypted Wi-Fi traffic. By intentionally injecting physical motion stimuli into the camera's environment and passively sniffing the resultant bursty traffic metadata, an attacker can accurately infer the camera's subscription level, operational state, and whether the owner is actively viewing the live stream. Through robust methodology, our evaluations successfully uncovered

these hidden states across various market-leading camera brands without requiring network decryption or authentication. This finding breaks the conventional assumption that standard encryption sufficiently safeguards the operational privacy of Consumer IoT devices.

In our second work, we shifted our focus to the physical vulnerabilities of analog sensing hardware, introducing *PhantomMotion*. This study demonstrated that the Passive Infrared (PIR) sensors universally utilized by battery-powered cameras as their primary wake-up component are critically vulnerable to targeted thermal and optical energy injections. By employing low-cost, off-the-shelf lasers, we proved that an adversary could reliably spoof human motion signatures from substantial distances. Crucially, we extended this attack beyond traditional line-of-sight boundaries to encompass non-line-of-sight (NLOS) environments. By directing lasers at opaque materials within the camera's field of view, the natural process of thermal diffusion generates a sufficient heat signature to trigger the camera's recording mechanism remotely and covertly. The *PhantomMotion* attack effectively bypasses physical barriers, allowing adversaries to initiate false alarms, actively manipulate the camera's attention, and evade detection without leaving a forensic digital footprint.

Collectively, these two bodies of work underscore a critical architectural disconnect in the design of modern smart surveillance systems: the reliance on differential network traffic profiles for economic monetization (subscription tiers), and the implicit trust placed in unverified, single-modality analog sensors. Addressing the vulnerabilities exposed in this dissertation will require an interdisciplinary paradigm shift across the industry. Future Consumer IoT platforms must strive to bridge the gap between cyber and physical security by adopting intelligent, battery-aware traffic obfuscation architectures and deploying robust sensor fusion

algorithms capable of physically validating the environment before committing to high-integrity network actions.

Bibliography

[1] CC3220R, CC3220S, and CC3220SF SimpleLink Wi-Fi Single-Chip Wireless MCU Solutions, 2022. https://www.ti.com/lit/ds/symlink/cc3220s.pdf.

[2] DHT11-temperature and humidity sensor, 2022. https://components101.com/sensors/dht11-temperature-sensor.

[3] How do I set up local storage backups using my Arlo Base Station or SmartHub?, 2022. https://kb.arlo.com/1146857/How-do-I-set-up-local-storage-backups-using-my-Arlo-Base-Station-or-SmartHub.

[4] The C-based Firmware Patching Framework for Broadcom/Cypress WiFi Chips that enables Monitor Mode, Frame Injection and much more, 2022. https://github.com/seemoo-lab/nexmon.

[5] Airmon-ng [Aircrack-ng], 2023. https://www.aircrack-ng.org/doku.php?id=airmon-ng.

[6] BISS0001: Micro power PIR motion detector IC, 2023.

[7] HC-SR501 PIR motion detector, 2023. https://www.mpja.com/download/31227sc.pdf.

[8] OUI, 2023. https://standards-oui.ieee.org/.

[9] pigpio, 2023. https://github.com/joan2937/pigpio.

[10] Qualcomm QCACLD WiFi (Android) monitor mode. https://github.com/kimocoder/qualcomm_android_monitor_mode, 2023.

[11] Ring LLC - MAC address information, 2023. https://ouilookup.com/vendor/ring-llc.

[12] Ring Security Camera Product Comparison, 2023. https://support.ring.com/hc/en-us/articles/115004687606-Ring-Security-Camera-Product-Comparison.

[13] SQLite, 2023. https://www.sqlite.org/index.html.

[14] Abbas Acar, Hossein Fereidooni, Tigist Abera, Amit Kumar Sikder, Markus Miettinen, Hidayet Aksu, Mauro Conti, Ahmad-Reza Sadeghi, and Selcuk Uluagac. Peek-a-boo: I see your smart home activities, even encrypted! In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '20)*, pages 207–218. Association for Computing Machinery, 2020.

[15] Fadel Adib, Zach Kabelac, Dina Katabi, and Robert C. Miller. 3d tracking via body radio reflections. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 317–329, Seattle, WA, April 2014. USENIX Association.

[16] Fabio Aiolli, Mauro Conti, Ankit Gangwal, and Mirko Polato. Mind your wallet's privacy: Identifying bitcoin wallet apps and user's actions through network traffic analysis. In *Proc. of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, pages 1484–1491. ACM, 2019.

[17] Aircrack-ng. Aircrack-ng - Main documentation, 2023. https://www.aircrack-ng.org/documentation.html.

[18] ALFA Network Inc. Awus1900, 2023. https://www.alfa.com.tw/products/awus1900?variant=36473966231624.

[19] Allied Market Research. Wireless video surveillance market, August 2022.

[20] Anas Alsoliman, Giulio Rigoni, Marco Levorato, Cristina Pinotti, Nils Ole Tippenhauer, and Mauro Conti. COTS drone detection using video streaming characteristics. In *International Conference on Distributed Computing and Networking 2021*, ICDCN '21, pages 166–175. ACM, 2021.

[21] Amazon. Alfa 1900. https://www.amazon.com/Alfa-AC1900-WiFi-Adapter-Long-Range/dp/B01MZD7Z76/, 2023.

[22] Ambarella. CV2S: Computer vision SoC for IP cameras, 2023.

[23] C.V Amrutha, C. Jyotsna, and J. Amudha. Deep learning approach for suspicious activity detection from surveillance video. In *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, pages 335–339, 2020.

[24] Apple Inc. Apple Developer Documentation: HTTP Live Streaming, 2022. https://developer.apple.com/documentation/http_live_streaming.

[25] Noah Apthorpe, Danny Yuxing Huang, Dillon Reisman, Arvind Narayanan, and Nick Feamster. Keeping the smart home private with smart (er) IoT traffic shaping. *Proc. on Privacy Enhancing Technologies*, 3:128–148, 2019.

[26] Noah Apthorpe, Dhanaanjoy Mondal, Mosharaf Chowdhury, and Nick Feamster. Pingpong: Packet-level signatures for smart home device events. *CoRR*, abs/1907.11797, 2019.

[27] Noah Apthorpe, Dillon Reisman, and Nick Feamster. Closing the blinds: Four strategies for protecting smart home privacy from network observers. *arXiv preprint arXiv:1705.06809*, 2017. Also appears in Workshop on Technology and Consumer Protection (ConPro '17).

[28] Noah Apthorpe, Dillon Reisman, and Nick Feamster. A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic. *arXiv preprint arXiv:1705.06805*, 2017.

[29] Arlo. What are Arlo's advanced motion alerts, and how do I set them up?, 2022. https://kb.arlo.com/000056585/What-are-Arlo-s-advanced-motion-alerts-and-how-do-I-set-them-up.

[30] Arlo. What is the maximum video resolution of Arlo cameras?, 2022. https://kb.arlo.com/1707/What-is-the-maximum-video-resolution-of-Arlo-cameras.

[31] Arlo. Annoying "alert" arlo secure notification. https://community.arlo.com/t5/Arlo-Pro-3-Floodlight/Annoying-alert-Arlo-secure-notification/td-p/1872974, 2023.

[32] Arlo. Arlo Go 2 LTE/Wi-Fi Security Camera, 2023. https://www.arlo.com/en-us/cameras/go/arlo-go-2-cameras.html.

[33] Arlo. Arlo secure: A command center for all your security, 2023. https://www.arlo.com/en-us/arlosecure.html.

[34] Arlo. Motion alerts are non stop at night. https://community.arlo.com/t5/Arlo-Pro/Motion-alerts-are-non-stop-at-night-Arlo-Pro/m-p/1196788, 2023.

[35] Arlo. What is continuous video recording (CVR) and how do I use it?, 2023. https://kb.arlo.com/1018425/What-is-continuous-video-recording-CVR-and-how-do-I-use-it.

[36] Sangwook Bae, Mincheol Son, Dongkwan Kim, CheolJun Park, Jiho Lee, Sooel Son, and Yongdae Kim. Watching the watchers: Practical video identification attack in LTE networks. In *31st USENIX Security Symposium*, pages 1307–1324. USENIX Association, August 2022.

[37] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *Proceedings of the Twenty-Third International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2938–2948, 2020.

[38] H. Barr and Human Rights Watch. *"My Life is Not Your Porn": Digital Sex Crimes in South Korea*. Human Rights Watch, 2021. https://www.hrw.org/report/2021/06/16/my-life-not-your-porn/digital-sex-crimes-south-korea.

[39] Bitmovin Inc. Bitmovin Video Developer Report 2021, 2022. https://go.bitmovin.com/video-developer-report-2021.

[40] Ravishankar Borgaonkar, Lucca Hirschi, Shinjo Park, and Altaf Shaik. New privacy threat on 3g, 4g, and upcoming 5g aka protocols. *Proceedings on Privacy Enhancing Technologies*, 2019(3):108–127, 2019.

[41] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, et al. Api design for machine learning software: experiences from the scikit-learn project. In *European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases*, 2013.

[42] Yulong Cao, S. Hrushikesh Bhupathiraju, Pirouz Naghavi, Takeshi Sugawara, Z. Morley Mao, and Sara Rampazzi. You can't see me: Physical removal attacks on lidar-based autonomous vehicles driving frameworks. In *32nd USENIX Security Symposium*, 2023.

[43] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[44] Yushi Cheng, Xiaoyu Ji, Tianyang Lu, and Wenyuan Xu. Dewicam: Detecting hidden wireless cameras via smartphones. In *Proc. of the 2018 on Asia Conference on Computer and Communications Security*, ASIACCS '18, pages 1–13. ACM, 2018.

[45] Yushi Cheng, Xiaoyu Ji, Tianyang Lu, and Wenyuan Xu. On detecting hidden wireless cameras: A traffic pattern-based approach. *IEEE Transactions on Mobile Computing*, 19(4):907–921, 2020.

[46] Donald J Coluzzi and A Robert. Laser fundamentals. *Principles and Practice of Laser Dentistry. St. Luis Missoiri2011*, pages 12–26, 2015.

[47] Mauro Conti, Luigi V. Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. Can't you hear me knocking: Identification of user actions on android apps via traffic analysis. In *Proc. of the 5th ACM Conference on Data and Application Security and Privacy*, CODASPY '15, pages 297–304. ACM, 2015.

[48] Trisha Datta, Noah Apthorpe, and Nick Feamster. A developer-friendly library for smart home IoT privacy-preserving traffic obfuscation. In *Proc. of the 2018 Workshop on IoT Security and Privacy*, IoT S&P '18, pages 43–48. ACM, 2018.

[49] Drew Davidson, Hao Wu, Rob Jellinek, Vikas Singh, and Thomas Ristenpart. Controlling UAVs with sensor input spoofing attacks. In *10th USENIX Workshop on Offensive Technologies (WOOT 16)*, Austin, TX, August 2016. USENIX Association.

[50] AJ Dellinger. Wired vs. wireless security cameras: How to choose which is best for you, October 2022.

[51] Om Dev, Shanker Dayal, Ashish Dubey, and SM Abbas. Multi-layered textile structure for thermal signature suppression of ground based targets. *Infrared Physics & Technology*, 105:103175, 2020.

[52] Suresh K Duggal. *Building materials*. Routledge, 2017.

[53] ecobee. Smart thermostats, 2023. https://www.ecobee.com/en-us/smart-thermostats/.

[54] Sheena Lewis Erete. Protecting the home: Exploring the roles of technology and citizen activism from a burglar's perspective. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2507–2516. ACM, 2013.

[55] Fact.MR. PIR sensor market, November 2022.

[56] S. Fang, Y. Liu, and P. Ning. Mimicry Attacks Against Wireless Link Signature and New Defense Using Time-Synched Link Signature. *IEEE Transactions on Information Forensics and Security*, 11(7):1515–1527, 2016.

[57] S. Fang, Y. Liu, and P. Ning. Wireless Communications under Broadband Reactive Jamming Attacks. *IEEE Transactions on Dependable and Secure Computing*, 13(3):394–408, 2016.

[58] S. Fang, Y. Liu, W. Shen, H. Zhu, and T. Wang. Virtual Multipath Attack and Defense for Location Distinction in Wireless Networks. *IEEE Transactions on Mobile Computing*, 16(2):566–580, 2017.

[59] S. Fang, I. Markwood, and Y. Liu. Manipulatable Wireless Key Establishment. In *2017 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9, 2017.

[60] Song Fang, Yao Liu, Wenbo Shen, and Haojin Zhu. Where Are You from? Confusing Location Distinction Using Virtual Multipath Camouflage. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 225–236, Maui, Hawaii, USA, 2014.

[61] Song Fang, Ian Markwood, and Yao Liu. Wireless-Assisted Key Establishment Leveraging Channel Manipulation. *IEEE Transactions on Mobile Computing*, 20(1):263–275, 2021.

[62] Song Fang, Ian Markwood, Yao Liu, Shangqing Zhao, Zhuo Lu, and Haojin Zhu. No Training Hurdles: Fast Training-Agnostic Attacks to Infer Your Typing. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 1747–1760, New York, NY, USA, 2018.

[63] Song Fang, Tao Wang, Yao Liu, Shangqing Zhao, and Zhuo Lu. Entrapment for Wireless Eavesdroppers. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pages 2530–2538, 2019.

[64] Federal Trade Commission. How to secure your home security cameras, 2023. https://consumer.ftc.gov/articles/using-ip-cameras-safely.

[65] Ellis Fenske, Dane Brown, Jeremy Martin, Travis Mayberry, Peter Ryan, and Erik C Rye. Three years later: A study of mac address randomization in mobile devices and when it succeeds. *Proc. Priv. Enhancing Technol.*, 2021(3):164–181, 2021.

[66] Wireshark Foundation. Wireshark, 2023. https://www.wireshark.org/.

[67] Stacy Fritz and Michelle Lusardi. White paper:"walking speed: the sixth vital sign". *Journal of geriatric physical therapy*, 32(2):2–5, 2009.

[68] Jairo Giraldo, Ehab Sarkar, Alvaro A Cardenas, Michail Maniatakos, and Murat Kantarcioglu. A survey of physics-based attack detection in cyber-physical systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[69] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2022.

[70] Daniel Golightly. What is Arlo Secure? Everything You Need To Know, 2022. https://www.androidheadlines.com/what-is-arlo-secure-everything-you-need-to-know.html.

[71] Jun Gong, Yang Zhang, Xia Zhou, and Xing-Dong Yang. Pyro: Thumb-tip gesture recognition using pyroelectric infrared sensing. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 553–563. ACM, 2017.

[72] Google. How Nest cameras store recorded video, 2023. https://support.google.com/googlenest/answer/9242083?hl=en.

[73] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. BadNets: Evaluating backdooring attacks on deep neural networks. In *IEEE Access*, volume 7, pages 47230–47244. IEEE, 2019.

[74] Ramakrishna Gummadi, David Wetherall, Ben Greenstein, and Srinivasan Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. In *Proc. of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '07, pages 385–396. ACM, 2007.

[75] Jianting Guo, Peijia Zheng, and Jiwu Huang. An efficient motion detection and tracking scheme for encrypted surveillance videos. *ACM Trans. Multimedia Comput. Commun. Appl.*, 13(4), sep 2017.

[76] X Guo, DK Tiller, GP Henze, and CE Waters. The performance of occupancy-based lighting control systems: A review. *Lighting Research & Technology*, 42(4):415–431, 2010.

[77] Xiaoxiao Guo, Shujian Cheng, Weiwei Cai, Yufeng Zhang, and Xue-ao Zhang. A review of carbon-based thermal interface materials: Mechanism, thermal measurements and thermal properties. *Materials & Design*, 209:109936, 2021.

[78] Neilly H. Tan, Richmond Y. Wong, Audrey Desjardins, Sean A. Munson, and James Pierce. Monitoring pets, deterring intruders, and casually spying on neighbors: Everyday uses of smart home cameras. In *CHI Conference on Human Factors in Computing Systems*, CHI '22. ACM, 2022.

[79] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. Do You Feel What I Hear? Enabling Autonomous IoT Device Pairing Using Different Sensor Types. In *IEEE Symposium on Security and Privacy (SP)*, pages 836–852, 2018.

[80] I. Haritaoglu, D. Harwood, and L.S. Davis. W/sup 4/: real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, 2000.

[81] Qiuye He and Song Fang. Phantom-csi attacks against wireless liveness detection. In *The 26th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, Oct 2023.

[82] Qiuye He, Song Fang, Tao Wang, Yao Liu, Shangqing Zhao, and Zhuo Lu. Proactive anti-eavesdropping with trap deployment in wireless networks. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 20(1):637–649, 2023.

[83] Qiuye He, Edwin Yang, and Song Fang. A Survey on Human Profile Information Inference via Wireless Signals. *IEEE Communications Surveys Tutorials*, pages 1–1, 2024.

[84] Qiuye He, Edwin Yang, Song Fang, and Shangqing Zhao. Honeybreath: An ambush tactic against wireless breath monitoring. In *EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous)*, 2022.

[85] Qiuye He, Edwin Yang, Song Fang, and Shangqing Zhao. Revisiting wireless breath and crowd inference attacks with defensive deception. *IEEE/ACM Transactions on Networking (ToN)*, 32(6):4976–4988, Dec 2024.

[86] Yan He, Qiuye He, Song Fang, and Yao Liu. Motioncompass: pinpointing wireless camera via motion-activated traffic. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '21, page 215–227, New York, NY, USA, 2021. Association for Computing Machinery.

[87] Yan He, Qiuye He, Song Fang, and Yao Liu. When free tier becomes free to enter: A non-intrusive way to identify security cameras with no cloud subscription. In *ACM Conference on Computer and Communications Security (CCS)*, Nov 2023.

[88] Yan He, Qiuye He, Song Fang, and Yao Liu. Precise wireless camera localization leveraging traffic-aided spatial analysis. *IEEE Transactions on Mobile Computing (TMC)*, 23(6):7256–7269, June 2024.

[89] Yan He, Guanchong Huang, and Song Fang. Phantommotion: Laser-based motion injection attacks on wireless security surveillance systems. In *Network and Distributed System Security (NDSS) Symposium 2026*, 2026.

[90] Yan He, Hanyan Zhang, Edwin Yang, and Song Fang. Virtual step pin pad: Towards foot-input authentication using geophones. In *IEEE International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, Dec 2020.

[91] Jeongyoon Heo, Sangwon Gil, Youngman Jung, Jinmok Kim, Donguk Kim, Woojin Park, Yongdae Kim, Kang G. Shin, and Choong-Hoon Lee. Are there wireless hidden cameras spying on me? In *Proc. of the 38th Annual Computer Security Applications Conference*, ACSAC '22, pages 714–726. ACM, 2022.

[92] Tin Kam Ho. Random decision forests. In *Proc. of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[93] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[94] Jack Philip Holman. *Heat transfer*. McGraw Hill, 1986.

[95] Guanchong Huang and Song Fang. Silhouettetell: Practical video identification leveraging blurred recordings of video subtitles. In *The 26th Privacy Enhancing Technologies Symposium (PETS)*, pages 1–16, 2026.

[96] Guanchong Huang, Yan He, Shangqing Zhao, Yi Wu, and Song Fang. Motiondecipher: General video-assisted passcode inference in virtual reality. In *The 28th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pages 298–313, 2025.

[97] Ignat Ignatov, Oleg Mosin, Hugo Niggli, Christos Drossinakis, and Georg Tyminski. Methods for registering non-ionizing radiation emitted from the human body. *European Reviews of Chemical Research*, 3(1):4–24, 2015.

[98] Intel. Intel Centrino Advanced-N 6200, Dual Band, 2022. https://www.intel.com/content/www/us/en/products/sku/59470/intel-centrino-advancedn-6200-dual-band/specifications.html.

[99] Keysight Technologies. Wireless analyzers, 2023. https://www.keysight.com/us/en/products/wireless-analyzers.html.

[100] Minhaj Ahmad Khan and Khaled Salah. Iot security: Review, blockchain solutions, and open challenges. *Future generation computer systems*, 82:395–411, 2018.

[101] Ludvig Kindberg. A frame differencing algorithm that allows for small camera movements., 2020.

[102] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[103] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology – CRYPTO 1999*, pages 388–397. Springer, 1999.

[104] Tom Kolnowski. Definitive Guide to Security Camera Subscription Plans, 2020. https://digitized.house/guide-to-security-camera-cloud-storage-plans/.

[105] Swarun Kumar, Ezzeldin Hamed, Dina Katabi, and Li Erran Li. Lte radio analytics made easy and accessible. *ACM SIGCOMM Computer Communication Review*, 44(4):211–222, 2014.

[106] Martin Kwan and Ken Fu. Regulating the sale and use of hidden cameras. *The Regulatory Review*, 2021.

[107] Ralph Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy*, 9(3):49–51, 2011.

[108] Laser Institute of America. The ANSI Z136.1 for safe use of lasers, 2022. https://www.lia.org/store/product/ansi-z1361-2022-safe-use-lasers-electronic-version.

[109] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[110] Jihyeon Lee, Sangwon Seo, Taehun Yang, and Soochang Park. Ai-aided hidden camera detection and localization based on raw iot network traffic. In *2022 IEEE 47th Conference on Local Computer Networks (LCN)*, pages 315–318, 2022.

[111] Ying Li, Yi Huang, Richard Xu, Suranga Seneviratne, Kanchana Thilakarathna, Adriel Cheng, Darren Webb, and Guillaume Jourjon. Deep content: Unveiling video streaming content from encrypted wifi traffic. In *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pages 1–8, 2018.

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

[113] Haili Liu, Ya Wang, Kevin Wang, and Hanbing Lin. Turning a pyroelectric infrared motion sensor into a high-accuracy presence detector by using a narrow semi-transparent chopper. *Applied Physics Letters*, 111(24):243901, 2017.

[114] Tian Liu, Ziyu Liu, Jun Huang, Rui Tan, and Zhen Tan. Detecting wireless spy cameras via stimulating and probing. In *Proc. of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '18, pages 243–255. ACM, 2018.

[115] Xuefeng Liu, Tianye Yang, Shaojie Tang, Peng Guo, and Jianwei Niu. From relative azimuth to absolute location: Pushing the limit of pir sensor based localization. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, MobiCom '20. ACM, 2020.

[116] Yanmao Man, Ming Li, and Ryan Gerdes. GhostImage: Remote perception attacks against camera-based image classification systems. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, pages 317–332. USENIX Association, 2020.

[117] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, 2007.

[118] Shrirang Mare, Franziska Roesner, and Tadayoshi Kohno. Smart devices in airbnbs: Considering privacy and security for both guests and hosts. *Proc. Priv. Enhancing Technol.*, 2020(2):436–458, 2020.

[119] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. A study of mac address randomization in mobile devices and when it fails. *Proc. on Privacy Enhancing Technologies*, 4:365–383, 2017.

[120] Jeremy Martin, Erik Rye, and Robert Beverly. Decomposition of mac address structure for granular device inference. In *Proc. of the 32nd Annual Conference on Computer Security Applications*, ACSAC '16, pages 78–88. ACM, 2016.

[121] Philip Matthews, Iljitsch van Beijnum, and Marcelo Bagnulo. Stateful nat64: Network address and protocol translation from ipv6 clients to ipv4 servers. RFC 6146, April 2011. https://www.rfc-editor.org/info/rfc6146.

[122] Tiffany May and Su-Hyun Lee. Is there a spy camera in that bathroom? In Seoul, 8,000 workers will check, 2018. https://www.nytimes.com/2018/09/03/world/asia/korea-toilet-camera.html.

[123] Stephen McLaughlin, Charalambos Konstantinou, Xiaoxiao Wang, Lucas Davi, Ahmad-Reza Sadeghi, Michail Maniatakos, and Ramesh Karri. The cybersecurity landscape in industrial control systems. *Proceedings of the IEEE*, 104(5):1039–1057, 2016.

[124] Stephen McLaughlin, Saman Zonouz, Devin Pohly, and Patrick McDaniel. A trusted safety verifier for process controller code. In *21st Annual Network and Distributed System Security Symposium (NDSS)*, 2014.

[125] Yan Meng, Haojin Zhu, Jinlei Li, Jin Li, and Yao Liu. Liveness Detection for Voice User Interface via Wireless Signals in IoT Environment. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[126] 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.

[127] Davide A Migliore, Matteo Matteucci, and Matteo Naccari. A revaluation of frame difference in fast and robust motion detection. In *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pages 215–218, 2006.

[128] Mostafa Mirshekari, Shijia Pan, Jonathon Fagert, Eve M. Schooler, Pei Zhang, and Hae Young Noh. Occupant localization using footstep-induced structural vibration. *Mechanical Systems and Signal Processing*, 112:77–97, 2018.

[129] Richard Mitev, Anna Pazii, Markus Miettinen, William Enck, and Ahmad-Reza Sadeghi. Leakypick: Iot audio spy detector. In *Annual Computer Security Applications Conference*, ACSAC '20, pages 694–705. ACM, 2020.

[130] Stefano Munari, Claudio E. Palazzi, Giacomo Quadrio, and Daniele Ronzani. Network traffic analysis of a small quadcopter. In *Proceedings of the 3rd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*, DroNet '17, pages 31–36. ACM, 2017.

[131] S.J. Murdoch and G. Danezis. Low-cost traffic analysis of tor. In *2005 IEEE Symposium on Security and Privacy (S&P'05)*, pages 183–195, 2005.

[132] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[133] Sujay Narayana, R. Venkatesha Prasad, Vijay S. Rao, T. V. Prabhakar, Sripad S. Kowshik, and Madhuri Sheethala Iyer. Pir sensors: Characterization and novel localization technique. In *Proc. of the 14th International Conference on Information Processing in Sensor Networks*, IPSN '15, pages 142–153. ACM, 2015.

[134] NETGEAR. AC2600 WiFi router (R7800), 2023. https://www.netgear.com/home/wifi/routers/r7800/.

[135] Phuc Nguyen, Hoang Truong, Mahesh Ravindranathan, Anh Nguyen, Richard Han, and Tam Vu. Matthan: Drone presence detection by identifying physical signatures in the drone's rf communication. In *Proc. of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '17, pages 211–224. ACM, 2017.

[136] TJ OConnor, William Enck, and Bradley Reaves. Blinded and confused: Uncovering systemic flaws in device telemetry for smart-home internet of things. In *Proc. of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '19, pages 140–150. ACM, 2019.

[137] International Bureau of Weights, Measures, Barry N Taylor, and Ambler Thompson. *The international system of units (SI)*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, 2001.

[138] Opteka. Opteka 650-1300mm: High definition telephoto zoom lens, 2023. https://opteka.com/products/op6501300.

[139] Eva Papadogiannaki, Constantinos Halevidis, Periklis Akritidis, and Lazaros Koromilas. Otter: A scalable high-resolution encrypted traffic identification engine. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 315–334. Springer, 2018.

[140] Eva Papadogiannaki and Sotiris Ioannidis. A survey on encrypted network traffic analysis applications, techniques, and countermeasures. *ACM Comput. Surv.*, 54(6), jul 2021.

[141] Jonathan Petit, Bas Stottelaar, Michael Feiri, and Frank Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. *Black Hat Europe*, 11(2015):995, 2015.

[142] James Pierce. Smart home security cameras and shifting lines of creepiness: A design-led inquiry. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 1–14. ACM, 2019.

[143] TQ Qiu and CL Tien. Heat transfer mechanisms during short-pulse laser heating of metals. *Journal of Heat Transfer (Transactions of the ASME (American Society of Mechanical Engineers), Series C);(United States)*, 115(4), 1993.

[144] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[145] Nipun Ramakrishnan and Tarun Soni. Network traffic prediction using recurrent neural networks. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, pages 187–193. IEEE, 2018.

[146] Reolink. How to turn on/off the PIR sensor, 2023. https://support.reolink.com/hc/en-us/articles/360004379493-How-to-Turn-on-off-the-PIR-Sensor.

[147] Ring. Ring community. https://community.ring.com/t/whats-expected-fps-for-spotlight-battery-cam-and-doorbell-2/19520, 2023.

[148] Ring LLC. Proper positioning for your floodlight cam, 2023. https://support.ring.com/hc/en-us/articles/360000124983-Proper-Positioning-for-Your-Floodlight-Cam.

[149] Kate Rooney. Booming subscription services are building revenue for companies but sapping customers' wallets, 2019. https://www.cnbc.com/2019/11/18/subscriptions-building-revenue-for-companies-but-sapping-wallets.html.

[150] James M Sabatier and Alexander E Ekimov. Ultrasonic methods for human motion detection. Technical report, Mississippi University National Center for Physical Acoustics, 2006.

[151] Muhammad Salman, Nguyen Dao, Uichin Lee, and Youngtae Noh. Csi: Despy: Enabling effortless spy camera detection via passive sensing of user activities and bitrate variations. *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(2):1–27, 2022.

[152] Sriram Sami, Sean Rui Xiang Tan, Bangjie Sun, and Jun Han. LAPD: Hidden spy camera detection using smartphone time-of-flight sensors. In *Proc. of the 19th ACM Conference on Embedded Networked Sensor Systems*, SenSys '21, pages 288–301. ACM, 2021.

[153] Domien Schepers, Aanjhan Ranganathan, and Mathy Vanhoef. Let numbers tell the tale: Measuring security trends in wi-fi networks and best practices. In *Proc. of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '21, pages 100–105. ACM, 2021.

[154] Daniel V Schroeder. *An Introduction to Thermal Physics*. Oxford University Press, USA, 2021.

[155] Matthias Schulz, Daniel Wegemer, and Matthias Hollick. Nexmon: A cookbook for firmware modifications on smartphones to enable monitor mode. *arXiv preprint arXiv:1601.07077*, 2015.

[156] Savio Sciancalepore, Omar Adel Ibrahim, Gabriele Oligeri, and Roberto Di Pietro. Pinch: An effective, efficient, and robust solution to drone detection via network traffic analysis. *Comput. Netw.*, 168(C), feb 2020.

[157] Scikit-learn. Scikit-learn. https://scikit-learn.org/stable/, 2023.

[158] Narmeen Shafqat, Cem Topcuoglu, Engin Kirda, and Aanjhan Ranganathan. Experience report on the challenges and opportunities in securing smartphones against zero-click attacks. *arXiv preprint arXiv:2211.03015*, 2022.

[159] Altaf Shaik, Ravishankar Borgaonkar, N. Asokan, Valtteri Niemi, and Jean-Pierre Seifert. Practical attacks against privacy and availability in 4g/lte mobile communication systems. In *Proc. 2016 Network and Distributed System Security Symposium*, 01 2016.

[160] Rahul Anand Sharma, Elahe Soltanaghaei, Anthony Rowe, and Vyas Sekar. Lumos: Identifying and localizing diverse hidden IoT devices in an unfamiliar environment. In *31st USENIX Security Symposium*, pages 1095–1112. USENIX Association, August 2022.

[161] Hocheol Shin, Dohyun Kim, Yujin Kwon, and Yongdae Kim. Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications. In *Proceedings of the 19th International Conference on Cryptographic Hardware and Embedded Systems (CHES 2017)*, pages 445–467. Springer, 2017.

[162] Akash Deep Singh, Luis Garcia, Joseph Noor, and Mani Srivastava. I always feel like somebody's sensing me! a framework to detect, identify, and localize clandestine wireless sensors. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1829–1846, 2021.

[163] Ashutosh Singh. How to root Android 13, 2022. https://www.ytechb.com/how-to-root-android-13/.

[164] Nishu Singla. Motion detection based on frame difference method. *International Journal of Information & Computation Technology*, 4(15):1559–1565, 2014.

[165] SpyGuy. Scout hidden camera detector, 2023. https://www.spyguy.com/products/scout-hidden-camera-detector.

[166] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. Light commands: Laser-Based audio injection attacks on Voice-Controllable systems. In *29th USENIX Security Symposium*, pages 2631–2648. USENIX Association, August 2020.

[167] Myunghoon Suk, Ashok Ramadass, Yohan Jin, and B. Prabhakaran. Video human motion recognition using a knowledge-based hybrid method based on a hidden markov model. *ACM Trans. Intell. Syst. Technol.*, 3(3), may 2012.

[168] Cameron Summerson. Rooting Android Just Isn't Worth It Anymore, 2021. https://www.howtogeek.com/335642/rooting-android-just-isnt-worth-it-anymore/.

[169] 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*, SPSM '15, pages 3–14. ACM, 2015.

[170] Vincent F. Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security*, 13(1):63–78, 2018.

[171] Teledyne FLIR LLC. Pro-grade thermal camera for smartphones: FLIR ONE pro, 2023.

[172] TensorFlow. TensorFlow, 2025. https://www.tensorflow.org.

[173] Birru Dereje Teshome. Spy camera epidemic in korea: a situational analysis. *Asian Journal of Sociological Research*, pages 1–13, 2019.

[174] Texas Instruments Incorporated. ADS1115, 2023.

[175] Kevin Sam Tharayil, Benyamin Farshteindiker, Shaked Eyal, Nir Hasidim, Roy Hershkovitz, Shani Houri, Ilia Yoffe, Michal Oren, and Yossi Oren. Sensor defense in-software (sdi): Practical software based detection of spoofing attacks on position sensors. *Engineering Applications of Artificial Intelligence*, 95:103904, 2020.

[176] Rahmadi Trimananda, Janus Varmarken, Athina Markopoulou, and Brian Demsky. Packet-level signatures for smart home devices. In *Network and Distributed Systems Security (NDSS) Symposium*, volume 2020, 2020.

[177] Timothy Trippel, Ofir Weisse, Wenyuan Xu, Peter Honeyman, and Kevin Fu. Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 3–18, 2017.

[178] TRUGLO. Trubrite 30 series. https://www.truglo.com/tru-brite-30-series/, 2023.

[179] Jennifer Pattison Tuohy. Why Get Home Security Cameras?, 2021. https://www.usnews.com/360-reviews/services/security-cameras/why-get-security-cameras.

[180] U.S. Department of Justice, Federal Bureau of Investigation. Ucr summary of reported crimes in the nation, 2024. Technical report, Uniform Crime Reporting (UCR) Program, Federal Bureau of Investigation, August 2025.

[181] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S. Cardoso, and Frank Piessens. Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms. In *Proc. of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, pages 413–424. ACM, 2016.

[182] Martin Vuagnoux and Sylvain Pasini. Compromising electromagnetic emanations of wired and wireless keyboards. In *18th USENIX Security Symposium*, pages 1–16, 2009.

[183] Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pages 36340–36357, 2023.

[184] Ju Wang, Jie Xiong, Hongbo Jiang, Kyle Jamieson, Xiaojiang Chen, Dingyi Fang, and Chen Wang. Low human-effort, device-free localization with fine-grained subcarrier information. *IEEE Transactions on Mobile Computing*, 17(11):2550–2563, 2018.

[185] Qi Wang, Pubali Datta, Wei Yang, Si Liu, Adam Bates, and Carl A. Gunter. Charting the attack surface of trigger-action iot platforms. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, pages 837–850, New York, NY, USA, 2019. Association for Computing Machinery.

[186] Douglas W Waples and Jacob S Waples. A review and evaluation of specific heat capacities of rocks, minerals, and subsurface fluids. part 1: Minerals and nonporous rocks. *Natural resources research*, 13(2):97–122, 2004.

[187] RHS Winterton. Newton's law of cooling. *Contemporary Physics*, 40(3):205–212, 1999.

[188] Tianna-Kaye Woodstock and Robert F. Karlicek. Rgb color sensors for occupant detection: An alternative to pir sensors. *IEEE Sensors Journal*, 20(20):12364–12373, 2020.

[189] Kevin Wu and Brent Lagesse. Do you see what i see? Detecting hidden streaming cameras through similarity of simultaneous observation. In *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom*, pages 1–10. IEEE, 2019.

[190] Chen Yan, Wenyuan Xu, and Jianhao Liu. Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle. In *DEF CON 24*, August 2016.

[191] Chen Yan, Zhijian Xu, Zhanyuan Yin, Xiaoyu Ji, and Wenyuan Xu. Rolling colors: Adversarial laser exploits against traffic light recognition. In *31st USENIX Security Symposium*, pages 1957–1974. USENIX Association, August 2022.

[192] Edwin Yang, Song Fang, Ian Markwood, Yao Liu, Shangqing Zhao, Zhuo Lu, and Haojin Zhu. Wireless training-free keystroke inference attack and defense. *IEEE/ACM Transactions on Networking (ToN)*, 30(4):1733–1748, Aug 2022.

[193] Edwin Yang, Qiuye He, and Song Fang. Wink: Wireless inference of numerical keystrokes via zero-training spatiotemporal analysis. In *ACM Conference on Computer and Communications Security (CCS)*, Nov 2022.

[194] Edwin Yang, Guanchong Huang, and Song Fang. Commandergps: Practical hidden command dissemination through pedal operations. In *2024 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9, 2024.

[195] Zhiyuan Yu, Zhuohang Li, Yuanhaur Chang, Skylar Fong, Jian Liu, and Ning Zhang. Heatdecam: Detecting hidden spy cameras via thermal emissions. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, pages 3107–3120. ACM, 2022.

[196] Feng Zhang, Chenshu Wu, Beibei Wang, Hung-Quoc Lai, Yi Han, and K. J. Ray Liu. Widetect: Robust motion detection with a statistical electromagnetic model. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 3(3), sep 2019.

[197] Zhaohe (John) Zhang, Edwin Yang, and Song Fang. CommanderGabble: A Universal Attack Against ASR Systems Leveraging Fast Speech. In *Annual Computer Security Applications Conference*, ACSAC '21, pages 720–731, New York, NY, USA, 2021.