

Noisy Networks, Nosy Neighbors: Simple Privacy Attacks Against Residential Wireless Traffic

Arne Roszeitis*, Bartosz Burgiel[†], Victor Jüttner*, Erik Buchmann*,

*Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, Leipzig University, Leipzig, Germany

Email: {arne.roszeitis, victor.juettner, erik.buchmann}@uni-leipzig.de

[†]DigiFors GmbH, Leipzig, Germany

Email: bartosz.burgiel@digifors.de

Abstract—Smart devices, such as light bulbs, TVs, fridges, etc., equipped with computing capabilities and wireless communication, are part of everyday life in many households. Previous work has already shown that a passive eavesdropper can derive private information, household routines, etc., from the network traffic of smart devices. However, existing attacks rely on capable adversaries with specialized machine learning expertise, labeled training data and reference devices, leaving it unclear how vulnerable ordinary households are to less sophisticated attackers. In this paper, we investigate the extent to which a “casual attacker” with straightforward IT skills and no specialized cybersecurity or ML tooling can reproduce such privacy attacks. Operating from an adjacent room in a real-world apartment building, we constrain our adversary to use only three off-the-shelf Raspberry Pis, Wireshark, and basic Python scripts. Through a three-week study, we demonstrate that this casual attacker can manually identify devices, recognize user states, track smartphone movements through walls via RSSI triangulation, and successfully extract detailed daily routines, including sleep patterns of guests. Our findings show that smart-home privacy leakage is a threat even from low-resourced, straightforward adversaries, e.g., neighbors.

Keywords—Smart Home, Privacy, Traffic Analysis, HAR

I. INTRODUCTION

Smart devices like light bulbs, TVs, and plugs have become a routine part of many households. They continuously communicate via wireless protocols like WiFi, Bluetooth Low Energy (BLE), or ZigBee for control, updates and cloud connectivity. Although the payload of the wireless communication is encrypted, the plain-text metadata of the protocols [1][2] allow privacy attacks. Using timing, packet sizes and radio characteristics, a passive attacker in an adjacent apartment can reveal which devices are present, when and where they are active, and ultimately the daily routines of the residents.

Prior work has shown that this threat is real and technically well-understood. Approaches, such as Peek-a-Boo [3], IoTBe-holder [4], WiFinger [5] and PingPong [6] demonstrate that encrypted smart-home traffic is sufficient to classify device types, infer device states, localize devices and perform Human Activity Recognition (HAR) on residents. However, all of these approaches assume a capable adversary: they require expert knowledge of machine learning pipelines, access to labeled training data or reference devices, and have been evaluated in carefully controlled smart-home testbeds.

What remains unclear is whether such attacks are also within reach of a casual attacker. We define a casual attacker as someone with a straightforward IT background, such as an undergraduate computer science student or a tech-savvy hobbyist. This attacker understands fundamental networking concepts and can write simple data-processing scripts, but lacks specialized tooling for cybersecurity or machine learning tasks. They operate with commodity hardware and have no prior access to the target’s floor plan or devices to collect training data. A motivated but technically modest attacker of this profile, such as a curious neighbor, presents a realistic threat. Therefore, we investigate the following research question:

Can a casual attacker reproduce smart-home privacy attacks in a real-world apartment?

To investigate this, we conduct a three-week study in a private apartment equipped with typical smart devices, with the consent of the residents. For this experiment, we assume the role of the casual attacker, eavesdropping on the apartment. To capture the WiFi and BLE traffic through the wall, the attacker is allowed to use only three Raspberry Pis. Furthermore, the subsequent data analysis is strictly limited to Wireshark and basic Python scripts for processing network packets and visualization. Using only these tools we attempt to achieve four privacy-invasive goals via a manual, heuristic pipeline: (1) device identification by combining publicly available MAC OUI (Organizationally Unique Identifier) lookups and BLE advertisements; (2) device state recognition (e.g., TV on/off) by observing basic traffic volume thresholds; (3) coarse indoor localization via basic RSSI triangulation from the three Raspberry Pis; and (4) behavior inference and guest detection by applying typical behavior patterns to plotted network activity over time.

We make two main contributions:

- We demonstrate that a casual attacker with limited resources and expertise can reproduce smart-home privacy attacks in an uncontrolled, real-world setting.
- We empirically detail this threat, showing the attacker can identify devices, recognize user states, track smartphone movements through walls, and extract daily routines including the presence and sleep schedules of guests.

Our findings demonstrate that smart-home privacy leakage is not limited to powerful adversaries with specialized equipment. This emphasizes the need for privacy mechanisms that

protect against close-by, passive, and persistent eavesdroppers of wireless metadata [1][7]. More detailed statistics and discussions can be found on a thesis [8].

Paper structure: Section II reviews related work. Section III outlines our research approach. The results of our experiments are shown in Section IV and then discussed in Section V. Finally, Section VI concludes the paper.

II. RELATED WORK

The related work provides state-of-the-art background as base for our attack pipeline and discusses existing smart-home traffic analysis and HAR attacks.

A. Background:

Traffic analysis and device discovery.: Encrypted smart-home traffic can be analyzed via metadata, such as IP/TCP headers, DNS queries and throughput. Apthorpe et al. show that an ISP-level observer, seeing only the gateway IP address, can infer device types by splitting traffic into substreams and inspecting DNS and rate information [2]. Device discovery can also exploit link-layer identifiers: OUIs in MAC addresses reveal manufacturers, and probe requests expose preferred Service Set Identifiers (SSIDs).

RSSI-based localization.: Received Signal Strength Indicators (RSSI) are widely used for indoor localization of wireless devices. RADAR demonstrates in-building RF-based user tracking using multiple WiFi receivers and signal strength [9]. Pérez Iglesias et al. apply Bluetooth RSSI fingerprinting to localize persons indoors with at least three receivers [10], and Mane et al. refine RSSI-based distance estimation in WiFi IoT settings using machine learning [11]. Precision depends on power, range and environment, but high accuracy is achievable in favourable conditions.

B. Smart-Home HAR and Traffic-Based Attacks

Passive network adversaries sniffing traffic from IoT devices are a known threat [1]. Acar et al. propose **Peek-a-Boo**, a multi-level attack that uses WiFi, ZigBee and BLE traffic to first identify devices, then infer their states and finally perform HAR on user activities [3]. Their evaluation is conducted in a controlled smart-home testbed and assumes access to similar devices for training and prior knowledge of the device set.

Zou et al. present **IoTBeholder**, which applies machine-learning pipelines to smart-home traffic and sensor data to build HAR models and predict user behaviour [4]. IoTBeholder integrates HAR deeply into the attack flow and targets habitual behaviours over time, but assumes expert-level data mining knowledge and carefully curated training datasets.

Trimananda et al. introduce **PingPong**, which learns packet-level signatures for smart-home device events using neural networks trained on labeled traces under WAN and WiFi sniffer threat models [6]. PingPong assumes a known device set and requires labeled reference traces for training.

Li et al. propose **WiFinger**, which aims to infer IoT device states from WiFi traffic and uses packet-level sequence matching to remain robust under packet loss and network

TABLE I. RELATED WORK ON SMART HOME PRIVACY ATTACKS AND HAR

Work	Attacker knowledge / resources
Peek-a-Boo [3]	Similar devices, labeled traces, multi-protocol capture, ML expertise
IoTBeholder [4]	Expert ML pipeline, large training datasets
WiFinger [5]	Prior device knowledge, access to similar devices, ML models
PingPong [6]	Trained neural networks, reference traces, known device set
Voice fingerp. [12]	Local network access, deep learning models
This work	Basic Python scripts, 3 Raspberry Pis

noise [5]. WiFinger operates under a WiFi sniffer threat model similar to that used above and still requires prior knowledge of the devices and access to similar devices for training machine-learning models.

Beyond general device and state inference, Wang et al. examine two voice-controlled devices and collect their encrypted traffic to infer the answers to voice prompts [12]. They use CNN, LSTM and stacked autoencoder models in an ensemble, assuming access to the local network and substantial machine-learning expertise. Collectively, these works show that encrypted smart-home traffic can be exploited for device-level and state inference, HAR and even content-related inference when attackers have access to suitable training data and controlled setups. Table I summarizes the attacker expertise in related work compared to our setting.

C. Research Gap

Existing work demonstrates that encrypted smart-home traffic can be exploited for device-, state- and activity-level inference, but typically assumes skilled adversaries, labeled training data and carefully controlled testbeds. Our study instead evaluates how far similar inferences are possible for a casual attacker using commodity hardware and simple scripts in a realistic household.

III. METHODOLOGY

In this section, we present our four-stage inference pipeline for the analysis, describe the experimental setup including the smart home environment and the monitoring station and detail the data collection procedure and pre-processing steps.

A. The Inference Pipeline

The inference pipeline is designed so that the casual attacker progressively builds knowledge about the smart home and its residents in four sequential steps:

- 1) **Device Identification:** First, we search for and identify network devices by analyzing MAC OUIs, BLE advertisements, and unencrypted setup traffic.
- 2) **Device State Recognition:** To determine each devices operational state (active, idle, or off), we monitor packet volumes over time and establish traffic thresholds.

- 3) **Device Localization:** We then aim to map physical locations. Using RSSI trilateration from three distributed sensors, we approximate the apartment's layout and track mobile devices through walls.
- 4) **Human Activity Recognition:** Finally, we synthesize this spatial and temporal data. By correlating smartphone movements with appliance states, we deduce sensitive daily routines like work schedules, sleep cycles, and guest presence.

B. Experimental Setup

The experiment was conducted in a five-room apartment equipped with a network of 10 to 12 wireless devices. Figure 1 illustrates the floor plan, while Table II lists the deployed devices and specifies their exact room assignments. This device mix reflects a modern household, consisting of automated smart home appliances, stationary multimedia and work devices, mobile personal smartphones, and a central WiFi router.

TABLE II. SMART DEVICES FOR OUR STUDY

Mac	Device	Conn.	Room
d8:f1	Tuya light bulb BKL1259	WiFi	1
08:b6	Shelly Plus HT	WiFi/BT	2
6c:5a	Tapo Bulb E225	WiFi	4
54:af	Tapo Smart Plug P100	WiFi/BT	4
8c:f6	Shelly Motion 2	WiFi	hall
9c:fc	Laptop (Intel)	WiFi/BT	1
24:2f	TP-Link router	WiFi	5
20:28	LG Smart TV UQ75009LF	WiFi/BT	5
60:1a	Nintendo Switch 2	WiFi	5
a4:45	Redmi Note 8 Pro	WiFi/BT	mobile
ae:90	Smartphone Guest 1	WiFi/BT	mobile
e2:e2	Smartphone Guest 2	WiFi/BT	mobile

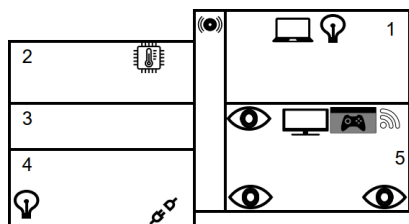


Figure 1. Floor plan of the experimental apartment: Office (1), Kitchen (2), Bath (3), Bedroom (4), and Living room (5).

To mimic the casual attacker operating from an adjacent room, we established a monitoring setup in the living room. Three Raspberry Pi 4 Model B units (4 GB RAM, PI OS Lite) equipped with TP-Link TL-WN722N WiFi antennas were placed in the corners of the room to enable RSSI triangulation. The adversary was restricted to standard software tools, specifically Wireshark [13], bluepy3 [14], and tcpdump [15]. These tools were controlled via simple Python scripts to capture WiFi

packets, BLE data, and RSSI readings without attempting to decrypt the payloads.

C. Data Collection and Pre-Processing

Data collection spanned a three-week period to capture a comprehensive picture of the resident's daily routines. This time frame included the initial factory-reset and installation of all smart devices, regular daily activities (e.g., leaving for university, leisure time at home), and a controlled evening visit by two guests carrying personal smartphones.

For pre-processing, the captured network traffic was categorized into three distinct datasets: general traffic data, signal strength vectors (RSSI), and probe request packets. To simulate the limitations of a casual adversary and to save storage space, all encrypted payloads were stripped from the traffic data, retaining only the payload length. Furthermore, both the traffic data and the RSSI values were aggregated into one-second intervals. The sanitized data was then visualized and analyzed using common Python libraries to execute the four objectives of our inference pipeline.

IV. STUDY RESULTS

Following our inference pipeline, we demonstrate how a casual adversary can gradually reconstruct a detailed picture of the residents of the apartment, their devices and their routines.

A. Device Identification

The first step is to isolate the target network, inventory the devices, and categorize their fundamental behavior.

a) *Passive Fingerprinting and Attribution:* By filtering network traffic based on signal strength, the adversary identifies frames belonging to the target apartment and thereby discovers its SSID. From there, all unique MAC addresses are extracted. Using simple Python scripts for OUI lookups [16][17], the attacker successfully attributes most smart appliances to their manufacturers (e.g., Tuya, TP-Link). Furthermore, captured BLE advertisements directly leak exact device models and names (e.g., "Complete Local Name: ShellyPlusHT-08B6"). Some smartphones and laptops utilize randomized MAC addresses that prevent direct OUI attribution, their presence is deduced from their complex, high-volume traffic.

b) *Device Installation Procedure:* The attacker gains a significant advantage when new devices are added to the network. During the coupling process, several smart appliances establish temporary, unencrypted WiFi access points. By eavesdropping on this setup traffic, the adversary extracts explicit device names (e.g., *Tapo E225*), firmware versions, and embedded web server traffic. Most alarmingly, the companion apps transmitted the household's primary WiFi SSID and password in plain text. Concurrently, the resident's connected smartphone leaked mDNS service names (e.g., Google, Spotify), immediately revealing the resident's digital ecosystem.

TABLE III. ATTACKER'S KNOWLEDGE AFTER DEVICE IDENTIFICATION.

MAC	Inferred Device (Source)	Profile	Mobility
d8:f1	Espressif Smart Device (MAC)	Auton.	Static
08:b6	Shelly Plus HT (BLE)	Auton.	Static
6c:5a	Tapo Bulb E225 (Installation)	Auton.	Static
54:af	TP-Link Smart Device (MAC)	Auton.	Static
8c:f6	Shelly Motion 2 (Installation)	Auton.	Static
9c:fc	Intel Multimedia (MAC)	Inter.	Static
24:2f	TP-Link Router (MAC)	Auton.	Static
20:28	LG TV UQ75009LF (BLE)	Inter.	Static
60:1a	Nintendo Switch (MAC)	Inter.	Static
a4:45	Redmi Note 8 Pro (Installation)	Inter.	Mobile

Note: Two guest smartphones excluded from baseline.

c) *Inferred Devices*: As summarized in Table III, the adversary successfully profiles the household's 10 baseline resident devices (the two guest smartphones are absent at this stage). Crucially, the table reflects the attacker's exact state of knowledge: while some devices leaked exact model names via setup traffic or BLE, others were inferred broadly via OUI manufacturer lookups (e.g., inferring a gaming console from a "Nintendo" MAC prefix). Despite varying levels of identification precision, mapping the traffic profiles (interactive vs. autonomous) and mobility states (static vs. mobile) provides the necessary foundation to track device states and physical locations in the subsequent phases.

B. Device State Recognition

By plotting packet counts over time, the attacker establishes simple heuristic thresholds to classify the operational state of a device as *active*, *idle*, or *off*.

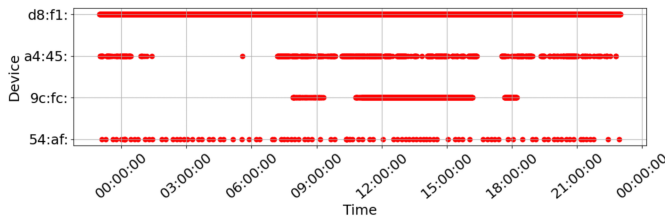


Figure 2. Network activity over one day.

As illustrated in Figure 2, observing network activity over 24 hours allows the attacker to classify devices into two distinct operational profiles based on their traffic signatures. First, *autonomous* devices, such as the Tuya light bulb (d8:f1) and the Tapo smart plug (54:af) exhibit continuous, periodic background traffic regardless of user presence. Second, *interactive* devices, such as the smartphone (a4:45) and the laptop (9c:fc) display dormant periods punctuated by sharp traffic spikes that directly correlate with active human usage. Additionally, by analyzing the variance in RSSI readings, the attacker distinguishes between *stationary* appliances (which exhibit minor signal fluctuations) and *mobile* devices. The

third and fourth column of Table III shows the profiles and mobility classes of the devices.

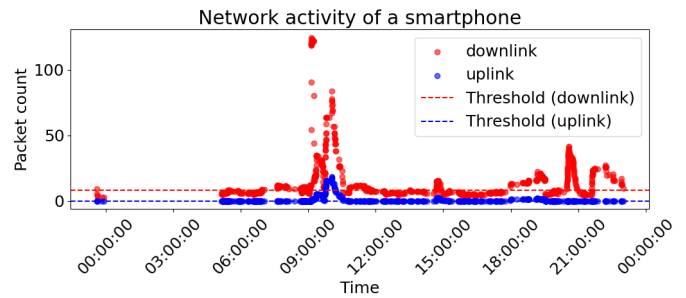


Figure 3. Network traffic thresholds of the interactive smartphone (a4:45).

The *interactive* multimedia devices provide highly legible state changes. As shown in Figure 3, the smartphone's traffic volume creates distinct thresholds: it remains off until 05:00, idles with minimal background syncs until 09:00, and then transitions into a clear, continuous active state for 1.5 hours. Similar, easily identifiable spikes were observed for the laptop and Smart TV (e.g., heavy downlink traffic spikes correlating with video streaming or boot-up sequences).

By applying these volume thresholds across the interactive devices, the adversary successfully maps the temporal rhythms of the resident's daily life. They now know *what* devices are in the home, and *when* the resident is actively using them.

C. Device Localization

With device identities and traffic profiles established, the attacker now aims to understand where these devices are located for a spatial grouping of devices into room-like regions.

We focus on the stationary devices as they can be used as anchors for room inference. To make this experiment more realistic with the obstruction by walls, we ignore all devices inside the living room. This leaves the five identified smart devices Tuya bulb (d8:f1), Shelly Plus HT (08:b6), Tapo bulb E225 (6c:5a), Tapo smart plug P100 (54:af), Shelly Motion 2 (8c:f6), and the laptop (9c:fc).

For these devices, we construct RSSI "fingerprints" by comparing the relative signal strengths received across the three spatially distributed Raspberry Pis. We interpret these fingerprints as 2D direction vectors pointing from the sniffing room towards each device. Figure 4 visualizes the resulting normalized direction vectors.

The vectors point toward the true locations for four of the six stationary devices. Shelly Plus HT (08:b6, gold) and Tapo bulb E225 (6c:5a, green) are misaligned, likely due to errors caused by wall density and electrical installations between them and the sniffers.

The attacker can still use these vectors to define 2-3 distinct sectors of the house. For example, the vectors for the Tuya bulb (d8:f1, blue) and the laptop (9c:fc, cyan) point into the same area, the attacker could label this as "office" area. Another area can be inferred by the positions of the Tapo bulb E225 (6c:5a, green) and Tapo smart plug (54:af, pink).

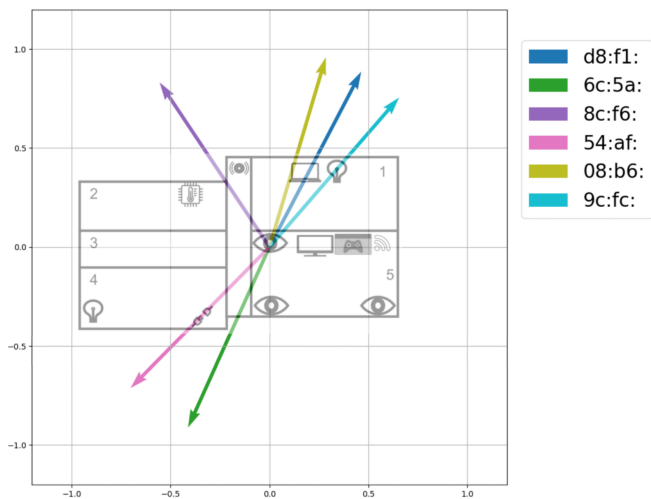


Figure 4. Normalized direction vectors of stationary devices based on relative RSSI. The floor plan is shown as ground truth for the reader; the attacker only observes the vectors.

Finally, the same directional mapping can be applied to the resident's smartphone. Over the course of a day, tracking the phone's changing direction vector allows the attacker to observe movement between these inferred regions. Some of these movement paths never intersect in the angular plots, strongly suggesting the physical presence of walls separating the inferred rooms.

D. Human Activity Recognition

In the final stage of the pipeline, the attacker synthesizes the device inventory, temporal state changes, and spatial mapping to reconstruct the resident's routines.

a) Weekday: Inspecting a randomly selected weekday, the attacker combines device states and localization to infer specific indoor activities.

- **Work from home:** Sustained laptop traffic originating from the inferred "work area," correlated with the smartphone remaining stationary in that same zone.
- **Breaks and Transitions:** Brief pauses in laptop activity coinciding with smartphone movement to the kitchen or bathroom area.
- **Evening Leisure:** A distinct shift occurs in the late afternoon. Laptop traffic ceases, and the Smart TV and Gaming Console become the dominant traffic sources in the living area, indicating the end of the workday.

This synthesis allows the attacker to confidently deduce the resident's daily rhythms, including wake-up times, working hours, and sleep cycles.

b) Weekly Schedule: By analyzing the presence and absence of interactive devices over a longer time frame (Figure 5), the attacker establishes macro-level routines. Rather than guessing specific external activities, the adversary reliably detects recurring periods of absence. For example, consistent, simultaneous drops in smartphone and laptop traffic on specific

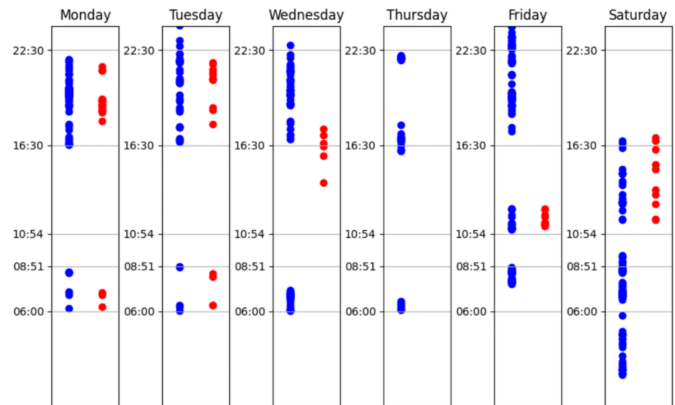


Figure 5. Weekly schedule over one week, showing network activity for the smartphone (blue) and laptop (red).

weekday mornings indicate absence, allowing the attacker to map the resident's standard out-of-home schedule.

c) Guest Visit: During the monitoring period, the inhabitant hosted two guests overnight. The attacker immediately detected this event when two unrecognized, interactive devices (smartphones) joined the network.

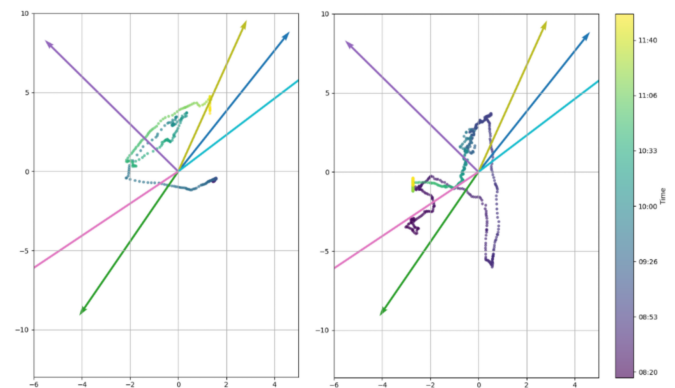


Figure 6. Location estimates of Guest 1 (left) and the inhabitant (right) during the evening.

By tracking these new devices alongside the baseline inventory, the attacker extracted a detailed timeline of the visit (Table IV). Observing the network traffic of the guest smartphones overnight allowed the attacker to infer the sleep cycles of the visitors. While the resident's and Guest 2's smartphones exhibited flat, minimal background traffic indicative of sleep between 02:00 and 08:00, Guest 1 generated traffic spikes at 04:00, indicating nighttime waking.

V. DISCUSSION

Our study demonstrates that a casual attacker can replicate the core privacy-invasive findings of prior work without relying on complex machine learning, curated training data, or reference devices. By applying basic heuristics to unencrypted metadata, we successfully extracted detailed daily routines and sleep cycles. This proves that the technical bar for a "nosy neighbor" attack is low. Ultimately, our goal is to inform

TABLE IV. INFERRED GUEST VISIT TIMELINE

Time	Identified Activity
21:00	Guest 1 arrives (new MAC appears)
22:00	Guest 2 arrives (new MAC appears)
01:30 – 02:00	Transition to sleep (interactive traffic drops)
04:00	Guest 1 wakes up (smartphone active)
08:00 – 09:00	Inhabitant and Guest 2 wake up
≈ 11:45	Guests leave (MACs disappear)

and warn users about how profoundly vulnerable their private networks are to simple, low-resource eavesdropping.

a) *The Role of the Smartphone vs. Smart Devices:*

Unlike most smart home privacy studies, our methodology goes beyond IoT appliances to analyze all wireless devices in the network, reflecting the behavior of a realistic adversary. Consequently, the resident's smartphone proved to be the most revealing source of information. While the smartphone indicates general user activity and movement, it is the smart home appliances that provide the necessary spatial anchors and semantic meaning. The specific type of a smart device immediately reveals what an area is used for, allowing the attacker to effectively profile the physical layout of the home. Furthermore, as other studies have shown, the traffic of these smart devices can be used directly to infer highly specific human activities. Finally, identifying sensitive smart home equipment, such as security cameras or microphones—allows an attacker to build an even more invasive profile of the house, potentially exposing security systems or hardware vulnerabilities. Thus, smart devices remain a critical component for an attacker aiming to comprehensively profile the home.

b) *Limitations of Simplistic Methods:* Our simplified approach does have limitations, particularly regarding device state recognition. While we easily inferred the states of interactive multimedia devices (like the TV and laptop) using simple traffic volume thresholds, this method failed to determine the operational states of autonomous smart appliances (e.g., the Tuya bulb or Shelly sensors). However, related work has proven that state inference for these devices is feasible using packet-level signatures. Therefore, this failure is strictly a limitation of our method, not a lack of underlying vulnerability. Finally, while our setup was realistically obstructed by an interior wall with electrical installations, a true cross-apartment attack might face heavier signal attenuation from walls or insulation.

VI. CONCLUSION

Our study has revealed that privacy risks in smart-home environments are not limited to highly capable attackers with specialized tools, training data, or machine learning expertise. Even a low-resourced adversary with inexpensive off-the-shelf hardware and simple analysis methods can infer sensitive information about residents' devices, activities, movements, and daily routines. By demonstrating these attacks in a re-

alistic apartment-building setting over an extended period, we show that the threat is both practical and relevant to ordinary households. This emphasizes the need for privacy mechanisms that also protect against close-by, passive, and persistent eavesdroppers of wireless network traffic.

REFERENCES

- [1] N. Aphorpe, D. Reisman, S. Sundaresan, A. Narayanan, and N. Feamster, "Spying on the smart home: Privacy attacks and defenses on encrypted IoT traffic," *ArXiv*, 2017.
- [2] N. Aphorpe, D. Reisman, and N. Feamster, "Poster: A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic," *Proc. NDSS*, 2016. [Online]. Available: <https://api.semanticscholar.org/CorpusID:15706672>.
- [3] A. Acar et al., "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*, ACM, 2020, pp. 207–218.
- [4] Q. Zou et al., "IoTBeholder: A privacy snooping attack on user habitual behaviors from smart home Wi-Fi traffic," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 7, no. 1, Mar. 2023.
- [5] R. Li, S. Liu, H. Hu, Q. Ye, and N. Feamster, "WiFinger: Fingerprinting noisy IoT event traffic using packet-level sequence matching," *ArXiv*, vol. abs/2508.03151, 2025.
- [6] R. Trimananda, J. Varmarken, A. Markopoulou, and B. Demsky, "PingPong: Packet-level signatures for smart home device events," *Proc. NDSS*, 2020. DOI: <https://dx.doi.org/10.14722/ndss.2020.24097>.
- [7] S. Wang, K. Yu, Q. Li, and D. Chen, "I still see you: Why existing IoT traffic reshaping fails," *ArXiv*, vol. abs/2406.10358, 2024. [Online]. Available: <https://api.semanticscholar.org/CorpusID:270560248>.
- [8] B. W. Burgiel, "Noisy networks, nosy neighbors: Inferring privacy invasive information from encrypted wireless traffic," *ArXiv*, Bachelor's Thesis, 2025.
- [9] P. Bahl and V. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings IEEE INFOCOM 2000*, vol. 2, 2000, 775–784 vol.2. DOI: 10.1109/INFOCOM.2000.832252.
- [10] H. J. Pérez Iglesias, V. Barral, and C. J. Escudero, "Indoor person localization system through RSSI Bluetooth fingerprinting," in *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2012, pp. 40–43.
- [11] S. Mane, M. Kulkarni, and S. Gupta, "RSSI-based indoor distance estimation in Wi-Fi IoT application using ai approaches," *International Journal of Communication Systems*, vol. 38, no. 5, 2025.
- [12] C. Wang et al., "Fingerprinting encrypted voice traffic on smart speakers with deep learning," *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020.
- [13] W. Foundation, *Wireshark*, Accessed: 2026-02-05. [Online]. Available: <https://www.wireshark.org/>.
- [14] I. Harvey, *Bluepy: Python interface to bluetooth LE on linux*, Accessed: 2026-02-05. [Online]. Available: <https://github.com/IanHarvey/bluepy>.
- [15] *tcpdump.org, Tcpdump - packet analyzer*, <https://www.tcpdump.org/>, Accessed: 2025-03-26.
- [16] *MAC Address Vendor Lookup*, <https://macaddress.io/>, Accessed: 2026-02-23.
- [17] *MA:CV:en:do:rs*, <https://macvendors.com/>, Accessed: 2026-02-23.