WILD Attack: Stealthy Undermining of Wi-Fi-Based Geolocation Through Remote Crowdsourced Data Injection

Changjia Zhu, Xiao Han*, Parush Gera, Zhuo Lu, Tempestt Neal, Yao Liu*
University of South Florida
{changjiaz, xiaoh, parush, zhuolu, tjneal, yliu21}@usf.edu

Abstract

Traditional Wi-Fi Positioning System (WPS) spoofing attacks, while seemingly effective, have failed to raise major WPS security concerns due to their lack of stealth and persistence. This paper introduces a novel *WILD Attack* that undermines WPS security by subverting its core infrastructure–the Location Lookup Table (LLT). In this attack, an adversary remotely submits falsified crowd-sourced reports for target Wi-Fi access points, inducing WPS providers to update LLT based on falsified rather than legitimate data. We examine four widely deployed WPS providers–Google, Apple, A-Map, and WiGLE–and observe that they all accept falsified reports and apply distinct policies to resolve conflicts between legitimate and falsified data. Exploiting these policies, the attacker can induce two forms of LLT subversion: *LLT Entry Tampering* and *LLT Entry Removal*, both persisting for weeks even after the attacker ceases activity. We further present three case studies that show the real-world impact of the *WILD Attack* and propose countermeasures to mitigate such threats.

1 Introduction

Wi-Fi Positioning Systems (WPS) serve as a valuable complement to GPS-based localization [4, 38, 42, 45], not only in environments where GPS signals are weak (e.g., indoors or dense urban areas), but also on devices that lack GPS hardware, such as desktop computers and IoT devices. When a device attempts to determine its location using WPS, it scans for nearby Wi-Fi access points (APs), collects their Basic Service Set Identifiers (BSSIDs), and sends this list to a WPS provider. The provider then queries its maintained database, commonly referred to as the Location Lookup Table (LLT), which maps each BSSID to its recorded geographic coordinates and returns the estimated location to the device.

Due to its critical role, WPS has been a target of spoofing attacks for decades [7, 10, 31, 37]. A classic WPS spoofing attack involves first collecting the BSSIDs of APs from a spoofed location—the site the attacker wants the victim device to believe it is in. The attacker then replays these BSSIDs at the victim's actual location by deploying rogue APs that impersonate legitimate ones. Meanwhile, the attacker jams signals emitted by legitimate APs at the victim's site so that victim devices receive only the rogue Wi-Fi signals. When these devices query WPS providers with the detected BSSIDs from rogue APs, the providers return the spoofed location, as those BSSIDs are mapped to spoofed coordinates in the LLT. Such attacks can therefore mislead location-based services.

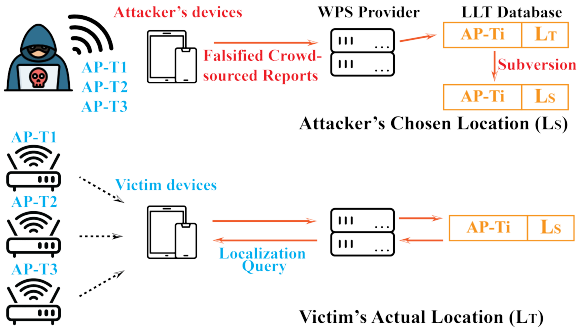


Figure 1: Scheme of the *WILD Attack*.

Prior WPS spoofing attacks are insightful and inspiring (e.g., [11, 37]); however, they suffer from certain limitations, primarily due to their lack of stealth and persistence. Specifically, all such attacks require the attackers to be physically near the victim and emit anomalous Wi-Fi signals, which inevitably raise the victims' suspicion. Additionally, spoofing is only effective when the rogue signals are being broadcast; once they stop, victim devices quickly revert to reporting their correct location. Given that attackers have no knowledge of when victims will need localization services, such short-lived attacks often have a limited impact.

Toward this, we raise two key questions: (1) Is it possible to attack WPS remotely? In other words, can the attacker undermine victims' WPS-based localization without being physically near the victim? (2) Is it possible to sustain the

*Corresponding authors.

undermining with minimal effort? Ideally, the attack effect persists even after the attacker ceases all activities. To explore these questions, we shift our focus from physically setting rogue APs near victims—as traditional WPS spoofing attacks do—to attack the core WPS infrastructure, specifically the LLT, whose integrity is critical to WPS security.

Unlike traditional databases, which are maintained and updated through protected internal channels, WPS providers rely on crowd-sourced data to maintain or update their LLT databases. Each time a mobile device uses WPS for localization, it can contribute raw data that helps update the LLT. This introduces a potential, yet unexplored vulnerability: Can an attacker exploit the crowd-sourcing process to directly attack and subvert the LLT? If so, the attacker could remotely target the LLT entries associated with APs surrounding the victim, thereby undermining the victim's WPS-based localization.

To explore this vulnerability, we introduce a novel **W**PS **I**nfrastructure-**L**evel **D**isruption attack, referred to as ***WILD Attack***. As illustrated in Figure 1, the attacker first remotely acquires the BSSIDs of legitimate APs (e.g., AP-T1) surrounding the victim's actual location (referred to as target APs/BSSIDs). Then, at an attacker's chosen location, she broadcasts these BSSIDs by crafting forged Wi-Fi signals. Devices at this site, including attacker-controlled devices as well as nearby unwitting user devices, thus receive these forged signals and report them, along with the GPS coordinates of the attacker's chosen location, as falsified crowd-sourced data to WPS providers. The providers are induced to update LLT based on these falsified reports, mapping the target BSSIDs to the coordinates of the attacker's chosen location instead of the victim's actual location, thereby subverting the corresponding records. In this attack, no rogue APs or suspicious signals appear near the victim. When a device at the victim's actual location later sends BSSIDs from nearby APs to the WPS provider for localization, it receives an incorrect result due to the subverted LLT.

In the *WILD Attack*, the falsified data is submitted through real devices and the normal crowd-sourcing procedure, making it difficult for WPS providers to detect. Meanwhile, the attacker's chosen location can be any site under her control or accessible to her, which directly addresses our first question of whether WPS can be attacked without physical proximity. Furthermore, as WPS providers do not update LLT entries in real time, the subverted records can persist after the attacker ceases activity, answering the second question of whether continuous disruption is feasible with minimal ongoing effort.

To launch the *WILD Attack*, the attacker must overcome two key challenges. The first is understanding how WPS providers handle falsified reports: specifically, whether they are accepted and under what conditions they can induce the LLT to be updated with falsified rather than legitimate reports. This challenge arises because, when falsified reports for the target BSSIDs are submitted from the attacker's chosen location, those BSSIDs remain physically operational at the victim's actual location and are included in legitimate reports from nearby devices. Thus, WPS providers receive conflicting data for the same BSSIDs from distinct geographic locations. To understand LLT update semantics under this condition, we examine four representative WPS providers: Google WPS [8], Apple WPS [3], A-Map [1], and WiGLE [43]. We systematically confirm that all these providers accept falsified reports and apply distinct policies to resolve conflicts between simultaneous legitimate and falsified data. Exploiting these policies, the attacker can strategically manipulate falsified data reporting and induce two LLT subversion patterns: *LLT Entry Tampering*, where the target entries are overwritten with falsified coordinates; and *LLT Entry Removal*, where the target entries are eliminated. In particular, Google WPS exhibits *LLT Entry Tampering*, with recorded coordinates overwritten after 11 days of falsified reporting, while Apple WPS exhibits *LLT Entry Removal*, with all target entries eliminated within just 3 days. Notably, regardless of the pattern, the subverted records persist even after the attack ceases, with no recovery for at least 4 weeks. The other two providers display similar patterns, which confirms the general applicability and long-lasting impact of the *WILD Attack*.

The second challenge is remotely acquiring a complete set of BSSIDs surrounding the victim's actual location, which is essential to identify which LLT entries to subvert. Prior work shows that the Apple Geolocation API can return nearby BSSIDs when queried with a known one [29], but this capability alone is insufficient, as the attacker lacks any initial BSSID at the victim's site. To address this, we introduce a *Hop-by-Hop Expansion* strategy, where the attacker begins with a known AP (e.g., one under her control) and iteratively queries the Apple API to discover neighboring BSSIDs, progressively advancing toward the victim's actual location. However, this approach often reveals only a subset of the target BSSIDs. To achieve full coverage, we propose a complementary *Last-Hex Enumeration* strategy. With both strategies, we show that the attacker can remotely obtain over 90% of the target BSSIDs.

To demonstrate the impact of the *WILD Attack*, we present three real-world case studies. While modern localization systems typically combine GPS and WPS to determine a device's location [17, 25], we show that, regardless of GPS status, undermining WPS via the *WILD Attack* severely degrades localization integrity, contradicting the common belief that GPS always guarantees accurate positioning. Specifically, the *WILD Attack* can induce:

- **Large-scale Denial-of-Service (DoS):** Silently cripple localization at city scale, sending devices into persistent geolocation blackouts.
- **Ride-hailing Misleading:** Mislead ride-hailing (e.g., Uber, Lyft, and Robotaxi) by kilometers, leaving drivers and passengers stranded on opposite sides of a city.
- **GPS-less Device Poisoning:** Provide precise false coordinates to GPS-less devices, tricking online services to deliver hyper-localized content for incorrect locations.

The contributions of this paper are as follows: **(1)** We introduce the *WILD Attack*, a WPS infrastructure-level disruption exploiting the crowd-sourcing process, enabling stealthy and persistent undermining of WPS localization. **(2)** We systematically analyze the conflict-resolution policies of four major WPS providers and evaluate the *WILD Attack* against each, uncovering two LLT subversion patterns: *LLT Entry Tampering* and *LLT Entry Removal*. **(3)** We propose the *Hop-by-Hop Expansion* and *Last-Hex Enumeration* strategies, which enable the remote discovery of comprehensive target BSSIDs. **(4)** We illustrate the real-world impact of the *WILD Attack* through three case studies, including large-scale DoS, ride-hailing misleading, and service poisoning on GPS-less devices.

2 Threat Model

We consider a non-privileged attacker who aims to undermine the WPS-based localization of devices at the victim's actual location, such that the victim devices receive either a falsified location or no location result. We assume that this victim's actual location is surrounded by several legitimate Wi-Fi APs, each with an associated entry in the WPS provider's LLT, which enables devices in the area to perform WPS-based localization. As these APs remain operational at the victim's actual location, WPS providers continue to receive legitimate crowd-sourced reports reflecting their true coordinates.

Our proposed *WILD Attack* is designed to make this undermining stealthy and persistent. As shown in Figure 1, the attacker seeks to subvert the LLT entries of legitimate APs surrounding the victim's actual location (i.e., target APs). Subverting an LLT entry means either overwriting its recorded coordinates (i.e., tampering) or eliminating the entry entirely (i.e., removal). To do so, the attacker first obtains the BSSIDs of target APs. Then, at an attacker's chosen location, she broadcasts forged Wi-Fi signals using these BSSIDs and submits falsified crowd-sourced reports through her controlled devices or nearby unwitting user devices.

Given that legitimate and falsified reports for the same BSSIDs originate from different locations, their coexistence introduces conflicts. The attacker must therefore understand how WPS providers handle falsified data and update LLT with such conflicting data. She can then strategically manipulate the reporting process to induce providers to update LLT based on falsified data, rather than retaining the original record reinforced by legitimate data. By subverting most—or ideally all—LLT entries associated with the target APs, the attacker ensures that devices at the victim's actual location receive undermined WPS localization, either a falsified location or no result at all, even though they detect only trusted AP signals.

3 LLT Update Semantics

This section examines how WPS providers process crowd-sourced data to update their LLT databases. We focus on four widely adopted providers: Google WPS, Apple WPS, A-Map (i.e., Gaode Maps in China), and WiGLE. We aim to answer the following questions: (i) Do WPS providers accept falsified reports or filter them out? (ii) If falsified reports are accepted, how do WPS providers resolve conflicts between them and legitimate reports from different locations? This is essential to determine whether, and under what conditions, an attacker can override legitimate data to subvert the LLT.

Through experimentation, we uncover the LLT update semantics of WPS providers. Section 3.2 confirms that falsified reports can be accepted to update the LLT, establishing a prerequisite for LLT subversion. Section 3.3 reveals how WPS providers handle conflicts between legitimate and falsified reports for the same BSSIDs, and how such conflicts influence LLT updates. Section 3.4 shows that an attacker can manipulate the reporting process to cause providers to prefer falsified data over legitimate data. Based on these findings, we derive how the attacker carries out the *WILD Attack*. We ethically conduct the experiments to prevent any impact on real users and detail the discussion in Appendix *Ethical Considerations*.

3.1 Implementation Primitives

We begin by introducing the core primitives throughout the experiments, including the generation of falsified reports and the probing of LLT entries.

Falsified Reports Generation: One seemingly straightforward approach in submitting falsified reports is to reverse-engineer the inner reporting interface used by WPS providers to collect crowd-sourced data—identifying required fields such as BSSID and GPS coordinates—and directly submit crafted reports that match the expected format. However, with the adoption of certificate pinning and stricter source verification by modern WPS providers, such direct injection techniques have been largely impractical [27, 40].

Instead, we consider a more feasible approach: the attacker emulates an AP to broadcast forged Wi-Fi signals that advertise the BSSIDs of target APs. This can be accomplished using low-cost microcontrollers (e.g., ESP32) or software-defined radios (SDRs) [11], which are programmed to emit beacon frames that conform to the structure of authentic Wi-Fi signals and embed the target BSSIDs. These signals are received by attacker-controlled or potentially nearby unwitting user mobile devices, which follow the standard system behavior and submit crowd-sourced reports to WPS providers. Since these reports stem from genuine device-side data collection but contain falsified information (i.e., BSSIDs that should not exist near the attacker), we treat them as falsified reports.

LLT Entry Probing: To examine LLT update semantics, we submit crowd-sourced reports for selected test BSSIDs

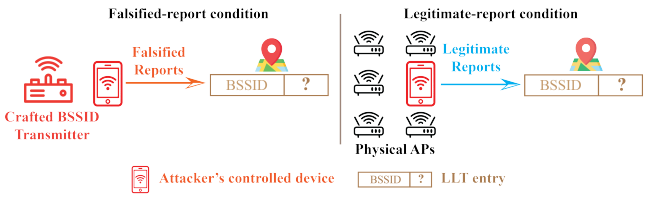


Figure 2: Legitimate vs. Falsified Reporting Conditions.

from one or multiple locations to WPS providers and observe how the corresponding LLT entries are updated. This requires the ability to precisely probe individual entries—specifically, given a BSSID, to verify whether it is recorded in the LLT and, if so, to retrieve its recorded coordinates. To achieve this, we leverage the location query APIs of WPS providers.

Modern WPS providers typically expose location query APIs to trusted third-party applications for their localization via observed BSSIDs. However, we find that they struggle to distinguish legitimate use from malicious probing. For example, although Apple does not officially expose its API, prior research has reverse-engineered the interface and documented its usage [2, 13]. Through empirical testing, we confirm that a non-privileged attacker can, in practice, access the location query APIs of all investigated WPS providers. By including the test BSSID in a query, one can probe its LLT status: if the BSSID is not recorded, the provider returns a null result; if it is recorded, the corresponding LLT entry is returned, e.g., `00:a2:ee:xx:xx:ad` $\rightarrow$ `(latitude, longitude)`.

3.2 Acceptance of Falsified Reports

Due to the black-box nature of WPS providers in handling crowd-sourced data, simply observing that an attacker device has submitted a report does not guarantee it is accepted. Thus, intercepting and analyzing traffic between mobile devices and WPS providers is ineffective, as their internal decision-making is opaque. To address this, we propose that, by monitoring whether falsified reports result in updates to the LLT entries, we can infer how WPS providers treat incoming reports.

Furthermore, by comparing the effectiveness (i.e., how often entries are updated) and efficiency (i.e., how quickly updates occur) between LLT updates induced by legitimate and falsified reports, we can assess whether WPS providers distinguish between them. If reports of equal volume lead to comparable LLT updates within similar time windows, it suggests that falsified and legitimate reports are treated equivalently.

Toward this, we design the experimental setup shown in Figure 2. In the falsified-report condition, we craft 5 BSSIDs that conform to the authentic BSSID format but are not associated with any physical APs and have never appeared in any WPS provider's LLT [14]. This ensures no legitimate crowd-sourced reports reference these BSSIDs. We then use a Wi-Fi microcontroller (ESP32) to emulate forged signals containing these BSSIDs at an attacker's chosen location, denoted as L_S. We configure three mobile devices to detect these signals and submit the observed BSSIDs—along with the GPS coordinates of L_S—as falsified reports to WPS providers.

WPS providers adopt two data collection methods: passive and active. Passive reports are automatically collected during the use of location-dependent services (e.g., Google Maps) [30], as seen in Google WPS, Apple WPS, and A-Map. Active reports are intentionally submitted by users through front-end interfaces [43], as in WiGLE. We select the appropriate application behavior to submit reports to each provider.

In the legitimate-report condition, we physically deploy 5 brand-new APs—each with a BSSID that has never appeared in any WPS provider's LLT—at the same L_S. The attacker-controlled mobile devices receive their signals and submit legitimate reports using the same operational procedures as in the falsified-report condition (e.g., identical devices and application behavior). Since the procedures are identical, we consider the report volumes to be similar in both conditions.

After continuously sending falsified reports and probing the LLT as discussed in Section 3.1, we observe that the attacker can successfully create forged records in the LLT of all investigated WPS providers with merely falsified reports. Specifically, for Google WPS, among the 5 target BSSIDs, it takes around 10 to 14 days for them to appear in the LLT with their coordinates recorded as L_S. For Apple WPS and A-Map, the required time is shorter—around 7 days. As for WiGLE, which supports active data submission, we find that LLT entries are created within 30 minutes of report processing.

Furthermore, under the legitimate-report condition, we observe that the recording time for the deployed APs is almost the same as that in the falsified-report condition. This suggests that WPS providers make no distinction between falsified and legitimate data, and that attacker-crafted falsified reports are equally effective and timely in updating the LLT.

3.3 Resolution Under Report Conflicts

When the attacker submits falsified reports for target APs at an attacker's chosen location, those APs continue to physically operate at the victim's actual location, where nearby devices observe and report their legitimate data. As a result, the WPS provider receives two sets of reports for the same BSSIDs from geographically distinct locations. We investigate how WPS providers handle this situation, specifically, whether these reports are treated as collaborative or competing.

Collaborative vs. Competing Criteria: To explore this, we design an experiment where crowd-sourced reports are submitted simultaneously from two locations, L_A and L_B. We fix L_A and gradually adjust the position of L_B, varying the distance between them from 50 to over 2,500 meters. At each site, we deploy an ESP32 device to emulate forged Wi-Fi signals using the same set of 5 BSSIDs that have not appeared in any

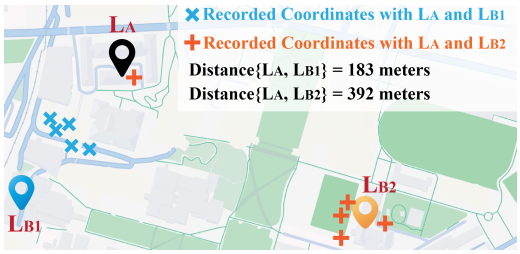


Figure 3: Effect of spatial distance between report locations on LLT updates.

WPS provider's LLT. We then follow the same operational procedures in Section 3.2 (identical devices and application behavior) at both locations simultaneously and monitor the resulting LLT entries. We observe that each WPS provider treats reports as collaborative when the spatial distance between reporting locations is within a certain threshold (e.g., approximately 200 meters for Google WPS), and as competing when the distance exceeds this threshold.

Taking Google WPS as an example, and as shown in Figure 3, when the distance between L_A and L_B (L_{B1}) is 183 meters, LLT entries are created obviously faster compared to submitting reports from a single location. Specifically, while reports from either L_A or L_{B1} alone take an average of 13 days to create LLT entries for the 5 target BSSIDs in Google WPS, simultaneous submissions from both locations reduce the time to 8 days. The recorded coordinates also fall between L_A and L_{B1}, averaging 105 meters from L_A and 91 meters from L_{B1}. This acceleration, along with the spatial averaging of coordinates, suggests the WPS provider aggregates reports from both sources and treats them as collaborative.

In contrast, Figure 3 also illustrates the case where the distance between L_A and L_B (L_{B2}) increases to 392 meters. In this setting, the LLT entries on Google WPS are created more slowly—averaging 23 days—and the resulting coordinates align exclusively with either L_A or L_{B2}. Specifically, among the five recorded coordinates for the target BSSIDs, four are near L_A and one near L_{B2}, with all falling within 20 meters of one of the two locations. This indicates that the WPS provider treats the reports as conflicting and does not aggregate them.

Through this experiment, we do not focus on the exact distance thresholds enforced by WPS providers. Instead, we reveal that when the attacker aims to subvert LLT remotely, the uncontrolled legitimate reports and their falsified reports are treated as competing. This raises a critical question: How does each provider handle such conflict? Specifically, which set of reports is trusted and ultimately used to update the LLT? Figure 3 indicates some LLT entries align with L_A, while others align with L_{B2}. To determine whether this outcome is random or governed by a consistent policy, we investigate the resolution strategies adopted by each WPS provider.

Conflict Resolution Hypotheses: Intuitively, we hypothesize that if the LLT update is deterministic, WPS providers may adopt one of several possible resolution policies when handling conflicting reports. (1) *majority-wins* policy, where the provider favors the location from which the greater number of reports originate; in this case, a high volume of reports from one location would outweigh a smaller volume from the other. (2) *last-report-wins* policy, where the provider prioritizes the most recently received data, allowing even a few falsified reports to override a larger volume of earlier legitimate ones. (3) *each-report-counts* policy, where all reports are considered equally and aggregated spatially, possibly leading to median-based coordinate updates that lie between conflicting locations. (4) *zero-trust* policy, in which conflicting data from distinct locations leads to the rejection of both sets of reports—either discarding the LLT entry or withholding updates until a clearer consensus is achieved.

We also consider the possibility that WPS providers employ implicit mechanisms [19,24], such as assigning trust scores to devices and giving greater weight to "trusted" entities when resolving conflicts. However, given the provider's reliance on data from billions of global devices, it is unlikely that individual user devices are explicitly prioritized. Even if such mechanisms exist, they are more likely limited to internally managed systems or approved partners, which fall outside the scope of a typical user or attacker in our investigation.

Resolution Logic Across Providers: To explore this, we extend the experimental setup in Figure 3 by systematically manipulating the spatial and temporal characteristics of conflicting reports. Specifically, we set L_A and L_B 2,500 meters apart, and define N_A and N_B as the number of attacker-controlled devices reporting from each site, where $0 \leq N_A, N_B \leq 10$. All devices exhibit identical application behavior to ensure consistency in the data submission process.

To differentiate between the hypothesized mechanisms, we vary both the relative number of devices (N_A vs. N_B) and the report submission order (e.g., sending fewer recent reports from one location after earlier reports from another). This allows us to observe whether WPS providers prioritize volume, recency, or reject updates entirely under conflicting conditions. To ensure precise control over the report submission, we isolate the area such that forged signals are not received by any unintended user devices.

Majority-wins: Towards Google WPS, we observe that when $N_A > N_B$, the LLT entries for the target BSSIDs are consistently recorded at L_A, and vice versa—when $N_B > N_A$, the entries align with L_B. This indicates a majority-wins policy, where the provider updates LLT entries based on the location that contributes the dominant number of reports. When $N_A = N_B$, we observe a result similar to that shown in Figure 3, where LLT entries are inconsistently assigned to either L_A or L_{B2}. We attribute this result to the asynchronous nature of Wi-Fi scanning on mobile devices: Wi-Fi signals are not captured continuously, and some BSSIDs may be intermittently missed, which introduces slight differences in report composition between locations even under identical conditions.

Furthermore, we find that this majority-wins policy also governs updates to existing LLT entries. Specifically, if an LLT entry is initially created based on a dominant number of reports from L_B (e.g., $N_A = 1, N_B = 4$), reversing the configuration such that $N_A > N_B$ (e.g., $N_A = 4, N_B = 1$) results in the LLT entry being updated and reassigned to L_A. This shows that Google WPS applies the majority-wins policy not only during initial entry creation but also when updating previously established LLT records in the presence of conflicting data.

Zero-trust: Apple WPS and A-Map adopt the zero-trust policy. Specifically, when these providers receive conflicting reports from distinct locations, they refuse to trust either source, even when one clearly contributes a higher volume of data. To demonstrate this, we deploy significantly different numbers of mobile devices at both locations (e.g., N_A = 10, N_B = 1). Despite continuously broadcasting signals and submitting reports for over a month, we observe that neither Apple WPS nor A-Map creates LLT entries for the target APs. This suggests that these providers reject LLT updates altogether when conflicting data is detected, regardless of report volume. Furthermore, in cases where the LLT entries for the target APs already exist, the introduction of conflicting reports leads to their swift removal. Specifically, Apple WPS removes the entries in approximately 3 days, while A-Map takes 2 days.

Furthermore, after removal, no new entries are recreated, indicating that the ongoing conflicting data prevents the reestablishment of the records. This may suggest a deliberate policy in Apple WPS and A-Map: when conflict arises from geographically distant locations, the system may infer that the BSSID belongs to a mobile hotspot (e.g., portable routers or tethering devices) rather than a stationary access point [32]. As a result, these providers appear to opt for eliminating such entries entirely rather than updating them with any reports.

Each-report-counts: In contrast to majority-wins and zero-trust policies, WiGLE adopts an each-report-counts policy, where every new report influences existing LLT entries. WiGLE employs an active report collection method requiring users to submit crowd-sourced data through its official front-end interface, so that the report volume can be precisely controlled. In our experiment, when a report is first submitted from L_A, LLT entries for the target APs are created within 30 minutes. Submitting another report from L_B then shifts the entries away from L_A toward L_B, and after about 10 reports from L_B, the entries fully align with L_B. We repeat this process in both directions and observe consistent outcomes: Each report contributes to the LLT update, but only after enough reports does the entry fully reflect the new reporting location.

3.4 Dominance of Falsified Data

As revealed in Section 3.3, under the majority-wins model, the attacker must submit enough falsified reports to override legitimate ones and thus dominate LLT update process. Beyond the intuitive approach of increasing the number of attacker-controlled mobile devices, we observe that she can also manipulate mobile application behavior to boost report volume.

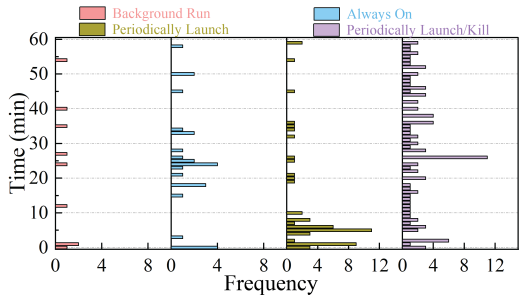


Figure 4: Temporal distribution of crowd-sourced reports under varying application behaviors.

We further examine Google WPS, which adopts the majority-wins policy. Google WPS uses a passive report collection method, where crowd-sourced data is gathered automatically during the use of location-dependent services. Taking Google Maps as an example, modern mobile systems allow it to run in the background. We monitored outbound traffic from the mobile device to the WPS provider over 60 minutes while Google Maps ran in the background, and observed that the frequency of crowd-sourced reporting was significantly low—about one report every 5 to 10 minutes, as shown in the "Background Run" condition of Figure 4.

When the user actively interacts with a location-dependent service—such as Google Maps—the device initially transmits location data more frequently, at about one report every 30-60 seconds. However, this elevated rate is temporary and gradually reverts to the background rate. As shown in Figure 4, we tested two user interaction modes that produce similar reporting patterns: "Always On," where the Google Maps remains continuously active, and "Periodically Launch," where the user alternates between foreground and background use.

We find that inducing device movement can prompt the application to continuously interact with the WPS provider, thereby sustaining high-frequency reporting. However, this conflicts with the attacker's goal of consistently submitting falsified reports with fixed GPS coordinates alongside the target BSSIDs. If the coordinates vary due to movement, the LLT entries cannot be updated with a stable location. To resolve this, we propose an automated scripting strategy. Specifically, the attacker deploys a lightweight script on the mobile device that periodically (e.g., every 2 minutes) launches and terminates location-dependent services (e.g., Google Maps). Each cycle re-initializes the service, keeping it in a fresh state that triggers location reporting, even though the device remains stationary. As shown in the "Periodically Launch/Kill" condition in Figure 4, this allows sending falsified reports with stable GPS coordinates at a significantly higher frequency, thereby enabling LLT subversion under majority-wins model.

3.5 Vulnerability Exploration

Building on the above findings, we demonstrate how an attacker can launch the *WILD Attack* against WPS providers with each class of conflicting-data resolution policy, thereby subverting the LLT entries associated with target APs.

Majority-wins policy: As illustrated in Figure 5, during the attack, the target APs remain active at the victim's actual location, and the attacker lacks precise knowledge of the volume of legitimate reports. For those WPS providers adopting the majority-wins policy (e.g., Google WPS), LLT entries can be subverted only if falsified reports outnumber legitimate ones. Thus, the attacker must maximize falsified report volume. To achieve this, she employs three key strategies:

Automated App Triggering: Following Section 3.4, the attacker applies the "Periodically Launch/Kill" behavior to her controlled devices, repeatedly launching and killing location-dependent services to sustain high-frequency reporting.

Unwitting Signal Reporting: Relying solely on attacker-controlled devices is limited, so the attacker leverages nearby user devices in high-traffic areas. She can select locations with dense mobile devices—such as train stations—as the attacker's chosen location L_S, where forged signals are broadcast. Mobile devices often run location-dependent services in the background by default, especially with location-enabled applications installed. As shown in Figure 4, such devices send reports even when services are not active, albeit at low frequency. When individuals pass through L_S, their devices detect forged signals and unwittingly submit falsified reports.

Collaborative Signal Broadcasting: Based on Section 3.3, reports from locations within a certain spatial threshold (e.g., 200 meters in Google WPS) are treated as collaborative and jointly induce the LLT updates. The attacker exploits this by deploying multiple ESP32 devices around L_S, each spaced within the identified spatial threshold and broadcasting the same target BSSIDs. Given the limited transmission range of Wi-Fi signals (typically a few dozen meters indoors), this distributed deployment allows more nearby user devices to detect the forged signals and unwittingly submit falsified reports.

Zero-trust policy: When WPS providers follow the zero-trust policy (e.g., Apple WPS and A-Map), the attacker does not need to maximize the volume of falsified reports. Instead, she can adopt any of the three strategies used for majority-wins policy to generate effective falsified submissions. Even a small number of falsified reports is sufficient to compete with legitimate data and trigger the removal of existing LLT entries. Moreover, due to zero-trust behavior, these providers do not recreate LLT entries once conflicting data is detected, even if falsified reports persist. Thus, the attacker can quickly shift to attack the next set of target APs once the current entries have been removed, thus significantly improving attack efficiency.

Each-report-counts policy: If WPS providers adopt an each-report-counts policy (e.g., WiGLE) or a last-report-wins policy (not observed in this study), the attacker can manipulate LLT entries immediately by submitting falsified reports. These reports shift the entries toward the attacker's chosen location L_S. She then monitors the entries for any subsequent shifts caused by new legitimate reports. If such shifts are detected, she can temporarily resume forged signals at L_S and resubmit falsified reports to re-subvert the entries. In the last-report-wins model, a single falsified report always suffices for subversion. In the each-report-counts model, if the attacker's objective is only to subvert existing entries without having to point to L_S, a single falsified report is also effective.

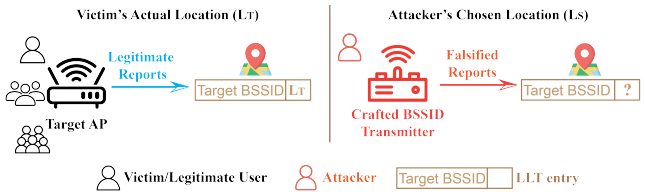


Figure 5: Experimental setup in WILD Attack evaluation.

4 WILD Attack

As mentioned, the *WILD Attack* aims to subvert LLT entries of legitimate APs surrounding the victim's actual location by submitting falsified reports that override or compete with legitimate data. Building on the vulnerability exploration strategies in Section 3, we now examine the feasibility of the *WILD Attack* in practical environments. Specifically, we assess the effectiveness of LLT subversion induced by a remote attacker (i.e., stealthiness) and the duration that the attack effects persist after falsified reporting ceases (i.e., persistence).

We selected three victim's actual locations (denoted as L_T) with different levels of ambient device traffic: a high-traffic zone (L_{Th}) on a university campus, a medium-traffic zone (L_{Tm}) in an urban residential complex, and a low-traffic zone (L_{Tl}) at a rural recreation center. Traffic density was measured by counting mobile devices within the target APs' signal reception range (approximately 50 m radius). Within this range, the high-, medium-, and low-traffic zones contained about 65, 24, and 7 devices, corresponding roughly to population densities of 10,000, 3,500, and 1,000 people/km^2, respectively. At each L_T, we selected 10 target APs, whose BSSIDs were already recorded in the LLTs of all investigated WPS providers. This allows us to examine whether falsified reports submitted at L_S by the attacker can compete with or override legitimate data of varying intensities.

When selecting the attacker's chosen location (L_S), we did not need to consider unrelated legitimate APs that may also exist around L_S, as we observed that they have minimal impact on the attack. Reports from L_S include both the unrelated legitimate BSSIDs at L_S and the target BSSIDs. Providers rely on the GPS coordinates embedded in these reports to

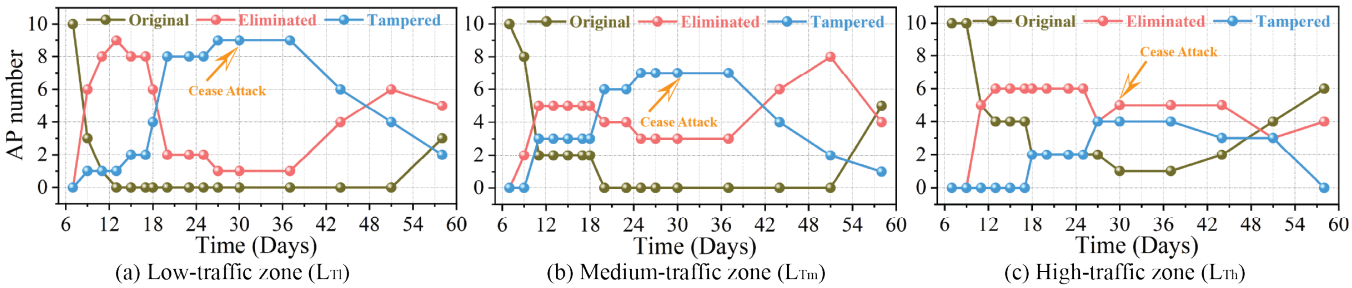


Figure 6: LLT subversion results for Google WPS (Day 0: start the attack activity; Day 30: cease the attack activity).

independently maintain or update BSSID-location mapping, keeping unrelated legitimate BSSIDs at their correct location (L_S) without affecting the handling of target BSSIDs. In our investigation, the L_T-L_S distances were around 11, 5, and 1 km for the low-, medium-, and high-traffic zones, respectively.

Through our investigation, we observed two LLT subversion patterns: *LLT Entry Tampering*, where a target BSSID's recorded location is overwritten with falsified coordinates, and *LLT Entry Removal*, where the corresponding entries are eliminated. Since WPS-based localization relies on accurate BSSID-to-coordinate mappings, both patterns severely disrupt accuracy. Moreover, the distortion persists for weeks after falsified reports cease. Table 1 summarizes the key subversion patterns of the investigated providers. Once most surrounding APs' records are subverted, victim devices are consistently misled or unable to obtain any location for a long duration.

As discussed in Appendix *Ethical Considerations*, we selected only a small subset of APs in each L_T for evaluation. This ensured that the majority of legitimate APs continued to dominate the localization, thereby preventing nearby user devices from being misled. However, it is important to note that these ethical constraints significantly limited the demonstrated impact of the attack. A real-world attacker, unconstrained by such considerations, could subvert all APs surrounding L_T.

4.1 LLT Entry Tampering

Google WPS: As shown in Figure 6, Google WPS is vulnerable to the *WILD Attack* under different levels of legitimate reporting activity. During the attack, LLT entries transitioned among three states: S_1 (Original): correct coordinates of the victim's actual location L_T; S_2 (Eliminated): entry removed from the LLT; and S_3 (Tampered): coordinates overwritten to the attacker's chosen location L_S.

Two-phase Progression Pattern: We observed a consistent two-phase progression pattern during the LLT subversion: In the first phase, entries transitioned from S_1 (Original) to S_2 (Eliminated), suggesting that Google WPS removed existing mappings. In the second phase, the entries moved from S_2 to S_3 (Tampered), indicating acceptance of the falsified coordinates once they outnumbered the legitimate data.

This two-phase progression is shown in Figure 6a. Starting from Day 0 (the beginning of falsified reporting), we observed that between Days 7-13, the number of S_2 entries increased while those in S_1 declined. Then, between Days 13-20, the number of S_3 entries rose, replacing S_2 entries, which confirms the $S_1 \rightarrow S_2 \rightarrow S_3$ transition. Occasionally, we also observed a one-phase progression, where entries moved directly from S_1 to S_3 without a prior elimination step. For example, on Day 9 of L_{Tl}, one entry transitioned directly from L_T to L_S.

This difference in progression patterns can be attributed to the confidence score tied to each LLT record. Along with recorded coordinates for each BSSID, Google WPS also maintains a confidence score that reflects the recorded location accuracy [8]. Records with lower confidence can be more susceptible to immediate updates (i.e., directly $S_1 \rightarrow S_3$), while higher-confidence records often require prior invalidation before being overwritten (i.e., $S_1 \rightarrow S_2 \rightarrow S_3$).

Subversion Effectiveness: The speed and extent of LLT subversion varied with legitimate reporting levels. In the low-traffic L_{Tl}, falsified reports dominated quickly: by Day 11, 1 AP had reached S_3 and 8 entered S_2; by Day 30, 9 were in S_3 and 1 in S_2. In the medium-traffic L_{Tm}, subversion progressed more gradually; by Day 11, 8 APs had shifted to S_2 or S_3. Even in the high-traffic L_{Th}, impact was still significant: by Day 30, 9 APs were in S_2/S_3, with only 1 remaining in S_1. This reveals a clear trend: lower legitimate reporting experiences faster $S_1 \rightarrow S_2/S_3$ transitions and higher subversion rates.

Subversion Persistence: To evaluate the attack persistence, we ceased forged signals on Day 30 and monitored the affected LLT entries over time. In L_{Tl}, after 2 weeks, 6 entries

Table 1: LLT Subversion Patterns of WPS Providers

Provider	Resolution Policy	Pattern	Attack Time	Persistence
Google WPS	Majority-wins	Tampering	$\sim$ 11 days	4 Weeks
WiGLE	Each-report-counts	Tampering	$<$ 30 min	Months
Apple WPS	Zero-trust	Removal	3 days	$>$ 2 months
A-Map	Zero-trust	Removal	2 days	2–3 weeks

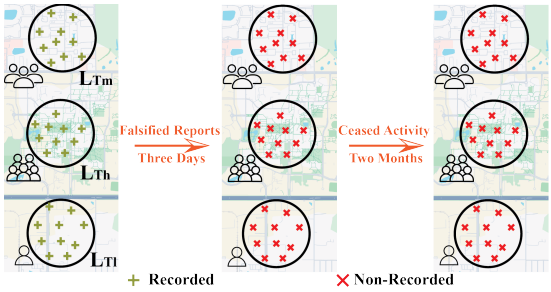


Figure 7: LLT subversion results for Apple WPS.

remained in S_3, while the rest remained in S_2. None reverted to their original state S_1. Similar persistence was observed in L_{Tm} and L_{Th}, with only two APs in L_{Th} recovering during the same period. Notably, even after 4 weeks, none of the target locations fully recovered. These results reveal that WPS suffers long-lasting trauma: once LLT entries are tampered with, the effect endures well beyond the end of attacker activity.

WiGLE: We observed that once a single falsified report was submitted to WiGLE, the LLT entries of target APs at any L_T shifted away from their original records, and with tens of falsified reports (the exact volume depending on the distance between L_S and L_T), the recorded coordinates were completely overwritten to L_S. After this initial subversion, we ceased falsified reports at L_S and monitored whether the entries would be corrected by legitimate reports. After 9 weeks, only 7 of the 10 APs in L_{Tm} showed entries that shifted slightly away from L_S due to legitimate reports, yet the updated coordinates still remained within about 80 meters of L_S. This demonstrates that user-contributed legitimate reports are relatively infrequent. Moreover, even when such reports exist, they are insufficient to restore the original LLT records.

4.2 LLT Entry Removal

Apple WPS: Apple WPS responded to falsified reports significantly faster than Google WPS. As shown in Figure 7, after only three days of falsified reporting from L_S, all target APs had their LLT entries eliminated—an abrupt and near-simultaneous effect across all APs. Notably, the results were consistent across all three L_T: regardless of legitimate report volume, falsified reports from L_S triggered complete removal of corresponding LLT entries within the same time window.

Even more concerning is the long-term impact of this removal. After the attacker ceased broadcasting forged signals, none of the eliminated LLT entries were restored for at least two months, despite our physical verification that signals from the target APs remained fully operational and continuously received by nearby user devices. This reveals a critical weakness: Apple WPS exhibits an *LLT Entry Removal* pattern, where even brief attack activity results in persistent data loss in the LLT, with no recovery for months.

A-Map: The *LLT Entry Removal* behavior in A-Map closely resembles that of Apple WPS, but with slightly faster responsiveness. We observed that just two days of continuous falsified reports from L_S were sufficient to eliminate the LLT entries associated with any L_T. While A-Map showed shorter attack persistence, the effect was still notable: after the attacker ceased activity, falsified entries for all 10 target APs in L_{Th} persisted for 13 days, and those in L_{Tl} for 19 days.

5 Target BSSIDs Discovery

As demonstrated in Section 4, the attacker can leverage the *WILD Attack* to remotely subvert the LLT entries of any target AP across all investigated WPS providers. To undermine the victim's WPS-based localization through the *WILD Attack*, the attacker needs to target the majority—if not all—of the APs surrounding the victim's actual location. Otherwise, the remaining unattacked legitimate APs may dominate the localization result. Since BSSID is the only information required to craft falsified reports, obtaining a complete set of target BSSIDs surrounding the victim's actual location is a critical prerequisite for undermining WPS-based localization.

The most straightforward way to obtain such information is for the attacker to physically visit the victim's actual location and scan nearby Wi-Fi signals. However, this contradicts the remote and stealthy nature of the attack and may be infeasible or costly. WiGLE offers a potential alternative by exposing coarse AP locations through its public database [33], but its utility is limited due to low accuracy and outdated entries from sparse contributors [10]. Prior work has shown that the Apple Geolocation API can return hundreds of nearby BSSIDs along with their coordinates when queried with a single BSSID [29]. Yet, this alone does not allow the attacker to directly obtain the target BSSIDs, since she does not initially know any BSSID within the victim's actual location. To address this, we propose a *Hop-by-Hop Expansion* strategy, which starts from any known AP and progressively expands toward the target APs.

5.1 Hop-by-Hop Expansion

Methodology: When queried with a single BSSID, the Apple Geolocation API returns not only its coordinates but also those of neighboring BSSIDs. We observe that the spatial coverage of these neighboring BSSIDs can extend up to 300 meters from the queried BSSID, far beyond its immediate vicinity (e.g., a 10-meter range). This broad spatial coverage enables the attacker to bridge significant geographic gaps. Specifically, leveraging this property, we propose that the attacker perform hop-by-hop queries: iteratively querying newly discovered BSSIDs that are progressively closer to the victim's actual location. This process gradually expands the attacker's knowledge of the AP environment towards the target and ultimately uncovers the target BSSIDs.

Figure 8: The hop-by-hop trace from Times Square, New York City, to a market near LaGuardia Airport (the number in each circle indicates the hop count in that area).

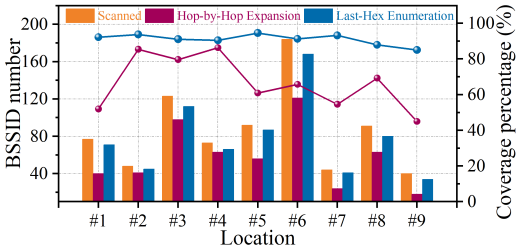


Figure 9: BSSID coverage by *Hop-by-Hop Expansion* and *Last-Hex Enumeration* compared to physically scanned list.

The BSSID discovery process begins with a single initial AP, either under the attacker's control or obtained from any sources. The attacker queries the Apple Geolocation API with this BSSID, receives nearby BSSIDs with coordinates, and sorts them by Euclidean distance to the victim's actual location. The nearest unqueried BSSID is then selected for the next query, and the process repeats. This hop-by-hop procedure progressively reduces the geographic gap between discovered BSSIDs and the victim's site. For example, as shown in Figure 8, an attacker starting in Times Square, New York, can discover the target BSSIDs near LaGuardia Airport (8.3 km away) in 59 hops.

When the initial AP is distant from the victim's actual location, the hop-by-hop strategy may encounter coverage gaps—regions with no discoverable BSSIDs due to sparse AP deployments—thereby preventing further progression. To address this, we incorporate a depth-first search (DFS) strategy that explores alternate branches of the AP graph when forward progress stalls. Specifically, if no newly discovered BSSID lies closer to the victim, the attacker backtracks and explores unvisited neighbors of previously queried APs. This recursive traversal ensures that, once a viable path toward the victim exists, it can eventually be identified. We summarize the *Hop-by-Hop Expansion* algorithm in Appendix A.

Evaluation: To assess whether the discovered BSSIDs represent all APs surrounding the victim's actual location, we selected nine sites in our city and applied the *Hop-by-Hop Expansion* to discover target BSSIDs. We then physically visited each site to scan nearby BSSIDs. After excluding those absent from all providers' databases (i.e., APs not recorded and thus irrelevant to localization or LLT subversion), the remaining scanned ones served as the full list of target BSSIDs at each site. As shown in Figure 9, *Hop-by-Hop Expansion* recovered on average 66.5% of the full list. For example, at Location #1, we physically scanned 77 BSSIDs, but only 40 were discovered, although the other undiscovered ones existed in Apple's database. This gap poses a challenge for attackers seeking to undermine WPS-based localization via the *WILD Attack*. To achieve comprehensive BSSID coverage, we introduce a complementary *Last-Hex Enumeration* strategy.

5.2 Last-Hex Enumeration

Methodology: To address the coverage gap, we conducted a closer inspection of the undiscovered BSSIDs. At Location #1, for example, we observed that 31 undiscovered BSSIDs differed only in the last hexadecimal digit from a corresponding discovered one. For instance, `78:29:ed:xx:xx:1f` was not discovered when `78:29:ed:xx:xx:1e` was. This pattern aligns with the common practice of a single physical AP broadcasting multiple BSSIDs across different frequency bands (e.g., 2.4 GHz and 5 GHz), where BSSIDs typically vary only in their final hexadecimal digit. We also find that the Apple Geolocation API often returns only one BSSID per AP while omitting others that share the same prefix.

Recognizing this pattern, we propose a *Last-Hex Enumeration* strategy. After collecting BSSIDs discovered through *Hop-by-Hop Expansion*, the attacker generates additional candidates by enumerating all possible variations of the last hexadecimal digit. For example, if `78:29:ed:xx:xx:1e` is discovered, the enumeration covers `78:29:ed:xx:xx:10` through `78:29:ed:xx:xx:1f`. These candidate BSSIDs are then queried directly via the Apple Geolocation API to confirm their presence and recorded coordinates. As the attacker only needs to enumerate at most 16 variations per discovered BSSID, and BSSIDs broadcasted by the same AP are typically consecutive (e.g., `78:29:ed:xx:xx:1e` and `78:29:ed:xx:xx:1f`), the additional overhead is minimal.

Evaluation: Applying the *Last-Hex Enumeration* to the BSSIDs obtained via *Hop-by-Hop Expansion* yields a more comprehensive list. As shown in Figure 9, nearly all locations achieve over 90% coverage, with an average of 91.1%. We note that, as the discovered BSSIDs encompass nearly all physically scanned ones, and any WPS provider can only record a subset—or at most the entirety—of physically scanned BSSIDs, the discovered list effectively reveals the comprehensive target BSSIDs, regardless of how each provider selects APs to record. Thus, these BSSIDs are sufficient for launching the *WILD Attack* to undermine WPS-based localization of any provider. We also consider the scenario where Apple stops returning neighboring BSSIDs as a defense against remote discovery of target BSSIDs. In this case, the discovered list of BSSIDs cannot be continuously updated.

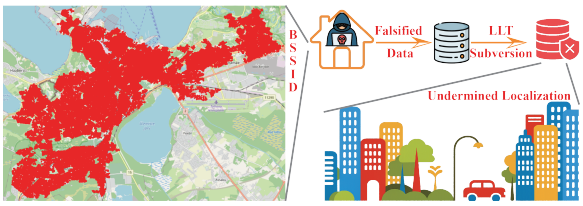


Figure 10: Visualization of a large-scale *WILD Attack*. Left: 262,073 BSSIDs discovered within a 10 km radius. Right: Illustration of large-scale WPS localization undermining.

However, we observed that the attacker's initially discovered BSSIDs would still represent the vast majority of those surrounding the victim's site for a prolonged period. A detailed discussion of this evaluation is provided in Appendix B.

6 Case Studies

In this section, we demonstrate the real-world impact of the *WILD Attack*. By subverting most—if not all—LLT records of target APs, the attacker undermines the victim's WPS-based localization. *LLT Entry Tampering* causes devices to receive falsified coordinates, while *LLT Entry Removal* erases location data. Consequently, if the undermined WPS output is adopted as the device's final location, any service relying on accurate geolocation would be misled or disrupted.

The WPS-based result does not always match a device's final location output. Modern localization systems typically combine GPS, WPS, and cellular signals, prioritizing the most accurate source [17, 22]. Since cellular positioning is less precise, typically a few hundred meters depending on cell tower density [22], the impact of the *WILD Attack* largely depends on GPS availability. While GPS is often assumed to always ensure accurate positioning, we show that it can be weak, unavailable, or absent on user devices, allowing undermined WPS to dominate localization. To illustrate this, we present three case studies under different GPS conditions: no signal, weak signal, and GPS-less devices. We then discuss why such impacts are impossible with traditional WPS spoofing.

6.1 Large-scale DoS

GPS jamming and interference are well-recognized threats. Beyond short-range jamming with low-cost noise transmitters such as USRP, high-power military-grade systems like Krasukha and Pole-21 [5, 44] can disrupt GPS signals across vast areas, extending hundreds of kilometers. Although designed for military use, these systems inevitably cause collateral interference to civilian GPS services.

Continuous monitoring from *GPSJAM.org* shows that several regions, particularly in Eastern Europe, face persistent GPS inference [9]. In such areas, where GPS signals are denied or unavailable, WPS becomes the primary source of civilian localization, making the *WILD Attack* especially potent. To illustrate the feasibility of a large-scale *WILD Attack* in GPS-denied zones, we select Tallinn, the capital of Estonia, as the victim's actual location. *GPSJAM.org* reports that Tallinn has consistently suffered strong GPS interference, likely due to its proximity to ongoing conflict zones and spillover from military operations. While Tallinn may not be a completely GPS-denied environment, such a condition is entirely plausible if GPS jamming operators intentionally target the area.

With the *Hop-by-Hop Expansion* and *Last-Hex Enumeration*, we discovered 262,073 BSSIDs within a 10 km radius centered at coordinates (59.437°N, 24.738°E)—the center of Tallinn, as shown in Figure 10. To avoid ethical concerns, we did not conduct actual LLT subversion in any WPS providers. As the feasibility of both *LLT Entry Tampering* and *LLT Entry Removal* has already been confirmed, we instead evaluate the cost and stealth of executing such a large-scale attack, exemplified by Google WPS and Apple WPS.

Assuming the attacker uses basic ESP32 units to broadcast forged signals at the attacker's chosen location, our experiments show that a single ESP32 device (costing about 1 USD) can simulate roughly 50 BSSIDs. To target 262,073 BSSIDs, the attacker would need about 5,242 ESP32 units, totaling 5,242 USD in hardware cost. She can then broadcast all forged signals using these devices at any her chosen location. As shown in Section 4, the attacker can swiftly eliminate all corresponding LLT entries from Apple WPS, effectively launching a large-scale Denial-of-Service (DoS) attack against the entire city, as shown in Figure 10. For Google WPS, the impact is even more severe, as the attacker can manipulate LLT entries to point to any designated locations.

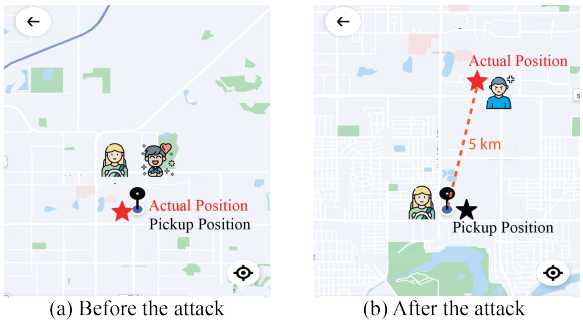
We note that this process is less likely to trigger filtering or firewall defenses from WPS providers, as the falsified reports are submitted by real devices within the normal operational flow. Meanwhile, the forged signals received by these devices are valid 802.11 frames broadcast over the air. Given the low cost and high stealth, such a large-scale attack is clearly feasible for ordinary attackers, let alone well-resourced adversaries such as state-sponsored or military actors.

6.2 Ride-hailing Misleading

Dense urban areas often experience weak GPS signals due to obstruction and multipath effects from surrounding buildings. To examine the impact of the *WILD Attack* under such conditions, we selected a victim's actual location with weak GPS signals and evaluated its effect on ride-hailing services, specifically Uber, the largest ride-sharing platform at the time of writing. We independently measured the localization accuracy of GPS and WPS: GPS alone provided an accuracy of about 115 meters, while WPS achieved 12 meters. Therefore, user devices prioritize WPS for its higher precision, as shown in Figure 11a, where the dark blue dot indicates the estimated location and the surrounding light blue circle represents the re-

(a) Before the attack (b) After the attack

Figure 11: (a) Accurate localization. (b) Offset localization and enlarged uncertainty radius. The small dark blue dot represents the device's estimated location, the surrounding light blue circle indicates the reported accuracy radius, and the black pin shows the pickup location, which typically matches the estimated location automatically.



(a) Before the attack (b) After the attack

Figure 12: Ride-hailing in the Uber app on an Android device.

ported uncertainty radius. This location aligns with the user's actual position and reflects the 12-meter accuracy of WPS.

As the *WILD Attack* induces *LLT Entry Removal* in Apple WPS and *LLT Entry Tampering* in Google WPS, we assessed the impact of each case on Uber ride-hailing. We carefully conducted the evaluation to avoid any harm to users and detail the precautions in Appendix *Ethical Considerations*.

LLT Entry Removal: After executing the *WILD Attack* against all surrounding APs at the victim's actual location, Apple WPS localization becomes unavailable. When victims open the Uber app on an iOS device to request a ride, their devices must rely solely on GPS, which is less accurate. As shown in Figure 11b, the estimated location deviates noticeably from the user's true position by around 60 meters compared to the precise WPS-based location in Figure 11a. Additionally, the surrounding light blue circle—representing the uncertainty radius—is around 100 meters, which is consistent with the accuracy of standalone GPS.

This discrepancy can cause substantial real-world inconvenience. When the user typically selects their localization output as the pickup point (indicated by the black pin), their actual position can be a hundred meters away, which is enough to span several city blocks in the dense urban area. The large uncertainty radius further increases the ambiguity of the pickup point. As a result, Uber drivers would struggle to find the passenger, leading to delays or even canceled rides.

LLT Entry Tampering: The *WILD Attack* causes a more severe impact on Google WPS by inducing misleading localization. Critically, the tampered APs can still yield higher accuracy than GPS signals. This is because WPS accuracy depends on the signal strength received by the device [4]. Although the LLT has been subverted, the signal strength of nearby APs remains consistent. Thus, the device obtains a high-confidence but incorrect localization from Google WPS, which takes precedence over the less precise GPS output.

We tampered with the LLT entries of target APs, relocating them 5 km from the original site, and then evaluated how Uber on an Android device responded. As shown in Figure 12, before the attack, Uber correctly identified the user's actual location. After the attack, however, it automatically selected a pickup point at the tampered location—5 km away—causing complete misalignment between the user and the driver. Notably, while this case study demonstrates a 5-km offset, the attacker's chosen location could be anywhere in the world, potentially resulting in even more disruptive consequences.

6.3 Service Poisoning on GPS-less Devices

Even when GPS signals are strong in the environment, devices without built-in GPS hardware, such as laptops and desktop computers, rely primarily on WPS for localization. Consequently, the *WILD Attack* can still cause significant harm to such devices. While one might assume WPS has minimal impact on such devices since they are not typically used for navigation or mapping, location services extend far beyond these purposes, with common uses including online shopping, targeted advertising, and personalized content delivery. In this case study, we show how *LLT Entry Tampering* in Google WPS can poison these services.

We selected an open-air zone with strong GPS signals as the victim's actual location, ensuring no other users were present. Then, we tampered with the LLT entries of surrounding APs, relocating them to Tokyo, Japan, and simulated a victim performing searches on a Lenovo Legion T7 desktop using the Google browser. As shown in Figure 13, location-related queries—such as "McDonald's" or "local facial spa"—returned results and advertisements for Tokyo rather than for the victim's actual location. This shows that even with strong GPS signals, the absence of GPS hardware on laptops or desktops forces devices to rely on subverted WPS data, which leads to mislocalized services.

Such subversion provides clear commercial incentives for potential adversaries; for instance, a business owner (e.g., a shopping mall) could subvert the LLT entries of APs in a competitor's vicinity to make her mall appear closer in "nearby" search results, effectively hijacking customer traffic from the

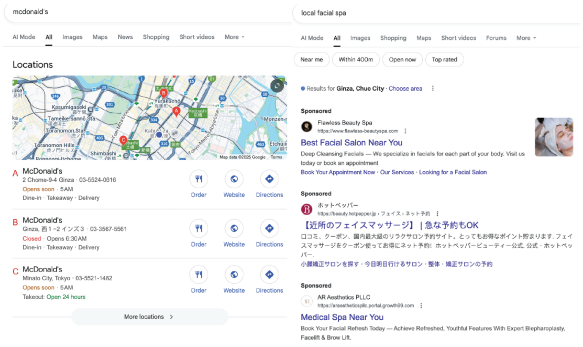


Figure 13: Location-related queries on desktop browsers.

competitor. We conducted the experiments under controlled conditions and detail our precautions in Appendix *Ethical Considerations*.

6.4 Why Existing WPS Spoofing Falls Short

With existing WPS spoofing attacks, it is infeasible to replicate the real-world impact demonstrated by the *WILD Attack*.

In the large-scale DoS scenario described in Section 6.1, the attacker targets a zone with a 10 km radius. With traditional WPS spoofing attacks, where each rogue AP covers only about tens of meters [18], achieving full spoofing signal coverage in the target zone would require deploying at least 40,000 spoofing devices uniformly across the area (assuming each device's signal spans 50 meters). Each device would need to continuously broadcast spoofing signals to maintain the attack, as victim devices would rapidly revert to accurate localization once the signals stop. From both logistical and economic perspectives, executing such an attack is infeasible.

In the ride-hailing misleading and GPS-less service poisoning scenarios, traditional spoofing faces two key limitations. First, the attacker has no knowledge of when the victim will engage with location-based services. Intuitively, it is infeasible to determine the exact moment a user opens a ride-hailing app to request a pickup. As the attacker cannot continuously track the victim and initiate spoofing at precisely the right time, effective spoofing is impractical. Second, the attacker lacks physical proximity to the victim. For example, if a victim uses a desktop at home to perform location-dependent web searches, it is unrealistic for the attacker to be physically present inside the victim's residence to initiate the spoofing.

7 Discussion and Countermeasures

This section discusses techniques to further amplify the attack impact, as well as countermeasures to mitigate the threat.

The *WILD Attack* considers that the attacker's chosen location for broadcasting forged signals coincides with her physical location. This ensures that falsified reports associate the target BSSIDs with the genuine GPS coordinates of the attacker's position, but it limits the selectable attacker's chosen location. However, this limitation can be readily overcome. By launching a GPS spoofing at her physical site—using software-defined radio (SDR) devices such as USRP—the attacker can induce nearby devices to receive falsified GPS signals corresponding to any site [16, 35, 36]. Thus, the falsified reports submitted by these devices will associate the target BSSIDs with attacker-assigned coordinates. Under the majority-wins, last-report-wins, or each-report-counts model, this allows LLT entries to be tampered with at any designated location without requiring her physical presence there.

To mitigate the risks of the *WILD Attack*, we first recommend that Apple WPS disable the feature of returning neighboring BSSIDs when queried with a single BSSID, or restrict responses to those within close proximity (e.g., 10 meters range). While Figure 14 shows that the target BSSIDs remain largely unchanged over a certain period, in the long run, this precaution would ultimately hinder the attacker's ability to obtain them remotely. Although the attacker could still physically visit the target area to perform Wi-Fi scanning, this approach is less scalable or stealthy.

Second, we find that the zero-trust model offers stronger protection against localization undermining. When conflicting data is detected, LLT entries are removed, preventing victims from receiving falsified output. While this may cause failed localization, it alerts users to abnormal behavior. By contrast, majority-wins, last-report-wins, and each-report-counts models continue providing service but silently deliver incorrect results, which are harder to detect. We thus encourage the adoption of zero-trust model when handling conflicting data.

Furthermore, we propose that the zero-trust model should also be implemented in a refined manner. Currently, Apple WPS and A-Map eliminate conflicting entries within two to three days, which may be overly aggressive. Slowing this elimination would reduce susceptibility to short-lived conflicts. Once conflicting reports cease, providers should also accelerate recovery by discarding outdated data and restoring correct entries based on recent legitimate reports.

Beyond restricting BSSID exposure and refining zero-trust policies, WPS providers could also integrate automated anomaly detection and verification mechanisms before applying LLT updates. For example, sudden or large-scale coordinate changes could trigger temporary quarantine of suspicious data until corroborated by independent signals such as GPS or cellular networks, when such signals are available, or by cross-checking against longer-term historical patterns in Wi-Fi reports. Providers might also assign higher credibility to reports from devices with stable, geographically consistent reporting histories or to locations corroborated by many independent devices, ensuring that one-off or highly anomalous submissions carry less influence. Finally, to reduce long-term disruption, automated recovery mechanisms could be introduced—for example, automatically restoring the most fre-

quently reported legitimate coordinates once anomalous data subsides, or reverting to the last known stable location if conflicting reports persist beyond a preset time window—so that correct location information is rapidly re-established without waiting for manual intervention.

Responsible disclosure: We responsibly disclosed our findings to all investigated WPS providers—Google, Apple, A-Map, and WiGLE—through their official vulnerability reporting channels, and provided detailed attack scenarios, reproduction steps, and proof-of-concept materials. Google acknowledged the report under its Bug Hunters program and initiated internal review by the Trust & Safety Team. Apple's Product Security Team likewise confirmed receipt and requested additional materials for verification under the Apple Security Bounty program. We are actively engaging with them to support issue reproduction and mitigation. A-Map recognized its value and stated that future product improvements will consider the vulnerability. WiGLE responded that its data integrity relies on per-user credibility and clustering algorithms to flag large-scale anomalies. However, our empirical observations show that even newly registered users can still induce effective subversion of WiGLE's location data, suggesting that the protection mechanisms may not fully prevent targeted LLT manipulation.

8 Related Work

WPS Spoofing Attacks: Tippenhauer et al. first introduced spoofing attacks against WPS, specifically targeting Skyhook [37]. Their study showed that by replaying AP information from a spoofed location at the victim's site and simultaneously jamming legitimate Wi-Fi and GPS signals, victim devices were misled into reporting falsified positions. Since then, jamming-assisted spoofing has been further investigated. Feng et al. demonstrated that other WPS providers, including Google, Apple, and Microsoft, are also vulnerable when rogue APs are combined with jamming of legitimate signals [7].

Jamming-based attacks face limitations as they rely on specialized hardware for physical-layer interference [12, 26, 39, 41, 46] and cause noticeable disruptions to wireless communication, making them highly detectable [15, 34, 47]. Several studies have therefore explored WPS spoofing attacks without jamming [10, 20, 23, 31]. Packetbridge demonstrated that deploying significantly more rogue APs than legitimate ones could ensure spoofing success [31]. A subsequent study further improved the spoofing effectiveness by reverse-engineering the localization algorithms [10], but the findings remain limited to Google WPS.

To our knowledge, the only pioneer work that discusses the integrity of WPS LLT database is by Tippenhauer et al. [37], which laid the foundation for later research. While [37] demonstrated the feasibility of injecting false data and corrupting LLT entries, its injection process appeared mainly as a by-product of the WPS spoofing, where the spoofing setup introduced new falsified entries for the internet access point AP_K. The subsequent data corruption presented in [37] focuses primarily on AP_K, without investigations on how providers process conflicting crowd-sourced reports at policy level. This may explain why follow-up research has largely focused on the WPS spoofing, leaving the infrastructure-level poisoning problem underexplored. Our paper advances this overlooked perspective through more in-depth and systematic investigations of WPS ecosystem. We explicitly analyze which policies providers adopt to handle conflicting crowd-sourced data and how attackers strategically exploit such policies.

Apple Geolocation API: Two prior works leveraged the Apple Geolocation API to gather Wi-Fi AP information. In IPvSeeYou, Rye et al. extracted MAC addresses from IPv6 addresses obtained via large-scale internet measurements, and mapped them to geographic locations using the Apple Geolocation API [28]. In a subsequent study, the same authors leveraged the feature of Apple Geolocation API that returns additional nearby BSSID coordinates beyond those explicitly queried [29]. This allowed them to collect globally BSSID locations and track the movement of APs over time.

In contrast, our work utilizes the Apple Geolocation API in a fundamentally different way. Rather than passively collecting BSSID–location mappings or analyzing AP mobility trends, we strategically target the APs in a specific victim's site. We propose two strategies that allow the attacker to start from any initial known AP and iteratively query newly discovered APs, progressively obtaining a complete set of BSSIDs near the victim rather than incomplete global AP mappings.

Crowd-sourcing in Location-Based Services: Many location-based services (LBS) rely on crowd-sourced reports, including mobility traces, traffic events, and Wi-Fi measurements, which creates an expansive attack surface for data poisoning. Wang et al. demonstrated that software-based Sybil devices can manipulate crowd-sourced mapping services on Waze and proposed defenses based on detecting suspicious co-presence relationships among reporters [40]. Similarly, Eryonucu showed that Google Maps' participatory-sensing signals can be distorted through forged sybil submissions, affecting crowd-derived indicators like traffic density [6].

Within the context of Wi-Fi positioning, Sapiezynski et al. analyzed the dynamics of maintaining accuracy in crowd-sourced wardriving [30]. While prior efforts, such as the work by Li et al. [21], proposed mechanisms to harden indoor positioning against malicious input, those defenses were primarily focused on local deployments. In contrast, our work shifts the focus toward the global WPS infrastructure. We investigate how the lack of cross-verification in provider-level update policies facilitates persistent and remote subversion of the location lookup table, moving beyond transient application-level disruption to long-term infrastructure poisoning.

9 Conclusion

This paper presents a novel *WILD Attack* that enables stealthy and persistent subversion of WPS infrastructure. We show that WPS providers adopt different resolution policies to handle conflicting data originating from different locations, which can be exploited to induce either *LLT Entry Tampering* or *LLT Entry Removal*, both of which undermine WPS-based localization. We also propose strategies for remotely acquiring complete target BSSIDs and demonstrate the real-world impacts of the *WILD Attack* through three case studies.

Acknowledgments

The authors would like to thank all anonymous reviewers for their insightful comments. This research was supported by the National Science Foundation (NSF) under grants 2044516, 2316719 and 2404741. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the NSF.

Ethical Considerations

The stakeholders involved in this study include WPS providers (Google, Apple, A-Map, and WiGLE), end users at L_S and L_T whose localization relies on WPS services, service providers depending on WPS data (i.e., Uber in our case study), and the broader WPS ecosystem that benefits from improved understanding and resilience of WPS infrastructure.

We consulted with our institution's IRB, which confirmed that this work does not constitute human subjects research. We took all necessary precautions to ensure that no real users were affected in any experiment, that no external services such as Uber were disrupted, and that risks to WPS providers were minimized. The following discussion details how these safeguards were applied to LLT update semantics, the WILD Attack investigation, and the case studies, as well as the resulting benefits to the WPS ecosystem.

LLT Update Semantics: In Section 3, we focus on understanding the semantics of LLT updates rather than executing a practical attack. To ensure ethical responsibility, all experiments were conducted using Wi-Fi APs fully under our control. During these experiments, each AP's LLT entry could exist in one of three states: (1) recorded with the correct physical location, (2) not recorded in the LLT, (3) recorded with a falsified location (i.e., a location differing from its actual physical position) due to falsified reports. In the first case, the corresponding LLT data poses no risks to users. In the latter two cases, once we observed that a correct coordinate had been eliminated or a falsified coordinate had been recorded, we powered off the corresponding APs to ensure that no other user devices could detect or interact with their signals, thereby preventing misleading the localization function on any users. Since these APs were deployed by us, powering them off did not affect the original AP environment or any real users.

WILD Attack: In Section 4, we examine the feasibility of the *WILD Attack* in practical environments. To prevent the experiments from impacting real users, we took all necessary precautions. First, we physically deployed our own Wi-Fi APs at the victim's actual locations. After the experiments, we retrieved all deployed APs to ensure that the attack activity was strictly confined to the APs under our control.

To prevent unintended WPS localization undermining (i.e., users receiving incorrect or no location results) at both the victim's actual location L_T and the attacker's chosen location L_S, we carefully designed our experimental setup. At L_T, we selected the placement of target APs to ensure they were surrounded by multiple legitimate, unattacked APs—typically one target AP accompanied by at least ten nearby legitimate APs. Based on our observations, even if the LLT entries of the target APs were subverted, the remaining legitimate APs dominated the localization process and prevented the nearby user devices from being misled.

At L_S, we leveraged user devices to contribute falsified reports by detecting forged signals. To avoid undermining WPS localization, we carefully limited the spatial density of forged signals, ensuring that surrounding legitimate APs remained dominant. Thus, user devices at L_S continued to obtain correct location results when contributing to the attack.

Case Studies: In the case studies, specifically Section 6.2 and Section 6.3, we leveraged the *WILD Attack* to target nearly all BSSIDs surrounding the victim's actual location. To ensure the evaluation did not affect any external users, we adopted the following procedures. First, we set up a set of self-controlled APs whose number exceeds that of the physical APs surrounding the victim's location. We then submitted falsified reports for these self-controlled APs to subvert the LLT records of their BSSIDs. During the subversion process, these APs were kept powered off, ensuring that no device could receive their signals and thus preventing any harm. Next, during closed hours when no external users were present at the victim's actual location, we powered on our self-controlled APs—strictly for less than five minutes—to construct a controlled environment in which the majority of surrounding BSSIDs were subverted. We then tested the impact on ride-hailing and location-based services on GPS-less devices. In the ride-hailing case study, we did not attempt to request an actual Uber pickup; instead, we only collected results from our test devices, ensuring no impact on Uber drivers. After completing the evaluation, we immediately powered off our self-controlled APs, ensuring that the subverted records no longer had any effect. Finally, we restored the subverted records of these self-controlled APs.

Furthermore, our study strictly adheres to the principles of the Menlo Report and follows the Terms of Service of WPS providers, while also offering benefits to the WPS community.

Minimizing Risks to WPS Providers: Our study also

ensured that risks to WPS providers were minimized. All usage of location query APIs was strictly compiled with the providers' Terms of Service and remained fully within the scope of normal WPS localization procedures. Furthermore, we did not attempt to exceed or bypass any request limits enforced by the providers. For APs whose LLT entries were subverted, we subsequently restored the correct records to the providers' LLT, ensuring that their databases remained accurate. Importantly, as discussed above, during the restoration period, these subverted entries had no adverse effect on any WPS users or on the normal operation of provider services.

Benefits to the WPS Ecosystem: Beyond minimizing risks, our study provides benefits to the community by systematically revealing how major WPS providers resolve conflicts between legitimate and falsified reports, and by exposing the vulnerabilities of their current policies. We further propose concrete countermeasures, including refined conflict-resolution models, that providers can adopt to resist such attacks. To the best of our knowledge, this work equips providers with actionable insights to strengthen their platforms and potentially protect billions of WPS users.

Open Science

We provide the complete set of artifacts used in this paper in a Zenodo repository: https://zenodo.org/records/17834160. The repository contains three core components: (1) LLT Probing Tools, which are used to probe whether and how LLT entries update; (2) Target BSSID Discovery Tools, which allow remote acquisition of the complete list of target BSSIDs. (3) BSSID–coordinate datasets used in the *WILD Attack*.

References

[1] A-map. A-map location-based services (LBS). https://lbs.amap.com/.

[2] François-Xavier Aguessy and Côme Demoustier. Rapport du projet de fin d'études: Interception des échanges dans une connexion ssl/tls - application à l'analyse des données de géolocalisation envoyées par un smartphone. Technical report, 2012.

[3] Apple. Apple location services and privacy. https://support.apple.com/en-us/102515.

[4] Roshan Ayyalasomayajula, Aditya Arun, Chenfeng Wu, Sanatan Sharma, Abhishek Rajkumar Sethi, Deepak Vasisht, and Dinesh Bharadia. Deep learning based wireless localization for indoor navigation. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–14, 2020.

[5] Deagel.com. Pole-21. https://www.deagel.com/Components/Pole-21/a003340, 2025.

[6] Cihan Eryonucu and Panos Papadimitratos. Sybil-based attacks on google maps or how to forge the image of city life. In *Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 73–84, 2022.

[7] Jun Feng, Liang Roy, and Guang Gong. Vulnerability analysis and countermeasures for Wi-Fi-based location services and applications. https://cacr.uwaterloo.ca/techreports/2014/cacr2014-25.pdf, 2014.

[8] Google. Google maps Wi-Fi positioning system. https://mapsplatform.google.com/maps-products/#geolocation.

[9] GPSJam.org. GPS interference map. https://gpsjam.org/, 2025.

[10] Xiao Han, Junjie Xiong, Wenbo Shen, Zhuo Lu, and Yao Liu. Location heartbleeding: The rise of Wi-Fi spoofing attack via geolocation API. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1383–1397, 2022.

[11] Adam Harvey. Skylift: Wi-Fi Geolocation Spoofing with the ESP8266. https://github.com/adamhrv/skylift, 2016.

[12] Naureen Hoque and Hanif Rahbari. Countering relay and spoofing attacks in the connection establishment phase of Wi-Fi systems. In *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, page 275–285, 2023.

[13] hubert3. isniff GPS. https://github.com/hubert3/iSniff-GPS/, 2023.

[14] IEEE. Mac address block large (ma-l). https://standards-oui.ieee.org/oui/oui.txt, 2023.

[15] Suman Jana and Sneha Kumar Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. In *Proceedings of the 14th ACM international conference on Mobile computing and networking (MobiCom)*, pages 104–115, 2008.

[16] Kai Jansen, Matthias Schäfer, Daniel Moser, Vincent Lenders, Christina Pöpper, and Jens Schmitt. Crowd-gps-sec: Leveraging crowdsourcing to detect and localize GPS spoofing attacks. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 1018–1031, 2018.

[17] Wook Rak Jung, Scott Bell, Anastasia Petrenko, and Anton Sizo. Potential risks of WiFi-based indoor positioning and progress on improving localization functionality. In *Proceedings of the Fourth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness*, page 13–20, 2012.

[18] Muhammad Faizan Khan, Guojun Wang, Md Zakirul Alam Bhuiyan, and Xu Li. Wi-fi signal coverage distance estimation in collapsed structures. In *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*, pages 1066–1073, 2017.

[19] Peng Li, Xiaotian Yu, He Xu, Jiewei Qian, Lu Dong, and Huqing Nie. Research on secure localization model based on trust valuation in wireless sensor networks. *Security and Communication Networks*, 2017(1):6102780, 2017.

[20] Tao Li, Yimin Chen, Rui Zhang, Yanchao Zhang, and Terri Hedgpeth. Secure crowdsourced indoor positioning systems. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1034–1042, 2018.

[21] Tao Li, Yimin Chen, Rui Zhang, Yanchao Zhang, and Terri Hedgpeth. Secure crowdsourced indoor positioning systems. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 1034–1042. IEEE, 2018.

[22] Lizzie Epton. Beyond GPS: Leveraging cell + WiFi triangulation for precise IoT location tracking. https://blues.com/blog/, 2025.

[23] Célestin Matte, Jagdish Prasad Achara, and Mathieu Cunche. Device-to-identity linking attack using targeted Wi-Fi geolocation spoofing. In *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec)*, pages 1–6, 2015.

[24] Chenglin Miao, Qi Li, Lu Su, Mengdi Huai, Wenjun Jiang, and Jing Gao. Attack under disguise: An intelligent data poisoning attack mechanism in crowdsourcing. In *Proceedings of the 2018 World Wide Web Conference (WWW)*, page 13–22, 2018.

[25] Kaveh Pahlavan and Prashant Krishnamurthy. Evolution and impact of Wi-Fi technology and applications: A historical perspective. *International Journal of Wireless Information Networks*, 28(1):3–19, 2021.

[26] Christina Pöpper, Nils Ole Tippenhauer, Boris Danev, and Srdjan Capkun. Investigation of signal and message manipulations on the wireless channel. In *European Symposium on Research in Computer Security (ESORICS)*, pages 40–59, 2011.

[27] Amogh Pradeep, Muhammad Talha Paracha, Protick Bhowmick, Ali Davanian, Abbas Razaghpanah, Taejoong Chung, Martina Lindorfer, Narseo Vallina-Rodriguez, Dave Levin, and David Choffnes. A comparative analysis of certificate pinning in Android & iOS. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC)*, page 605–618, 2022.

[28] Erik Rye and Robert Beverly. IPvSeeYou: Exploiting leaked identifiers in IPv6 for street-level geolocation. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 3129–3145, 2023.

[29] Erik Rye and Dave Levin. Surveilling the masses with Wi-Fi-based positioning systems. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 2831–2846, 2024.

[30] Piotr Sapiezynski, Radu Gatej, Alan Mislove, and Sune Lehmann. Opportunities and challenges in crowdsourced wardriving. In *Proceedings of the 2015 Internet Measurement Conference (IMC)*, pages 267–273, 2015.

[31] Bengt Sjölen and Gordan Savicic. Packetbridge: Wireless geographical network intervention. https://criticalengineering.org/projects/packetbridge/, 2014.

[32] Ye Su, Liang Chen, and Xiaoyan Liu. Crowdsourced WiFi fingerprint localization in urban canyon. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 46:177–184, 2022.

[33] Aju Mathew Thomas, Gowtham Akshaya Kumaran, R Ramaguru, R Harish, and K Praveen. Evaluation of wireless access point security and best practices for mitigation. In *5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT)*, pages 422–427, 2021.

[34] Yinghua Tian, Nae Zheng, Xiang Chen, and Liuyang Gao. Wasserstein metric-based location spoofing attack detection in WiFi positioning systems. *Security and Communication Networks*, 2021(1):8817569, 2021.

[35] Christopher Tibaldo, Harshad Sathaye, Giovanni Camurati, and Srdjan Capkun. GNSS-WASP: GNSS wide area spoofing. In *34rd USENIX Security Symposium (USENIX Security)*, 2025.

[36] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. On the requirements for successful GPS spoofing attacks. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS)*, page 75–86, 2011.

[37] Nils Ole Tippenhauer, Kasper Bonne Rasmussen, Christina Pöpper, and Srdjan Čapkun. Attacks on public WLAN-based positioning systems. In *Proceedings of the 7th international conference on Mobile systems, applications, and services (MobiSys)*, pages 29–40, 2009.

[38] Deepak Vasisht, Swarun Kumar, and Dina Katabi. Decimeter-level localization with a single WiFi access point. In *Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation (NSDI)*, page 165–178, 2016.

[39] Triet Dang Vo-Huu, Tien Dang Vo-Huu, and Guevara Noubir. Interleaving jamming in Wi-Fi networks. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec)*, page 31–42, 2016.

[40] Gang Wang, Bolun Wang, Tianyi Wang, Ana Nika, Haitao Zheng, and Ben Y. Zhao. Defending against sybil devices in crowdsourced mapping services. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, page 179–191, 2016.

[41] Xianglin Wei, Qiping Wang, Tongxiang Wang, and Jianhua Fan. Jammer localization in multi-hop wireless network: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 19(2):765–799, 2017.

[42] Yongyong Wei and Rong Zheng. Handling device heterogeneity in Wi-Fi based indoor positioning systems. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 556–561, 2020.

[43] WiGLE.net. Wireless geographic logging engine. https://wigle.net/.

[44] Wikipedia. Krasukha. https://en.wikipedia.org/wiki/Krasukha, 2025.

[45] Jie Xiong, Karthikeyan Sundaresan, and Kyle Jamieson. Tonetrack: Leveraging frequency-agile radios for time-based indoor wireless localization. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom)*, page 537–549, 2015.

[46] Wenyuan Xu, Wade Trappe, Yanyong Zhang, and Timothy Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, pages 46–57, 2005.

[47] Ayong Ye, Qing Li, Qiang Zhang, and Baorong Cheng. Detection of spoofing attacks in WLAN-based positioning systems using WiFi hotspot tags. *IEEE Access*, 8:39768–39780, 2020.

A Hop-by-Hop Expansion Algorithm

Algorithm 1 Hop-by-Hop Expansion

// L_T: Coordinates of the victim's actual location
// Q: Priority queue sorted by distance to L_T
// D: Set of discovered target BSSIDs
// V: Set of visited BSSIDs to avoid re-querying
// AppleQuery($bssid$): Returns set of (bssid, coord)
// Dist($coord_1, coord_2$): Euclidean distance between two coordinates
// WithinRange($coord, L_T$): True if distance $\leq$ 100m

$b_0 \leftarrow$ Initial known BSSID (under attacker's control or known from other sources)
Insert $(b_0, \perp)$ into Q
while Q is not empty **do**
  $(b, coord) \leftarrow Q$.pop()
  if $b \in V$ **then**
    continue
  end if
  $V \leftarrow V \cup \{b\}$
  $R \leftarrow$ AppleQuery(b)
  for each $(b', coord') \in R$ **do**
    if $b' \notin V$ **then**
      if WithinRange($coord', L_T$) **then**
        $D \leftarrow D \cup \{b'\}$
      end if
      Insert $(b', coord')$ into Q with priority Dist($coord', L_T$)
    end if
  end for
end while
return D ▷ Target BSSIDs Discovered

B Sustained Utility of Discovered BSSIDs

We further evaluate the condition that, if Apple immediately disables the feature of returning neighboring BSSIDs when queried with a single BSSID, how long the attacker's previously discovered BSSIDs remain effective without ongoing updates. Specifically, we assess, after a certain period, what proportion of the BSSIDs surrounding the victim's actual location had already been captured during the attacker's initial discovery. To conduct this evaluation, we first applied the proposed strategies to discover BSSIDs within a selected 1 km area, and then continuously repeated the same procedure to monitor changes relative to the initial list.

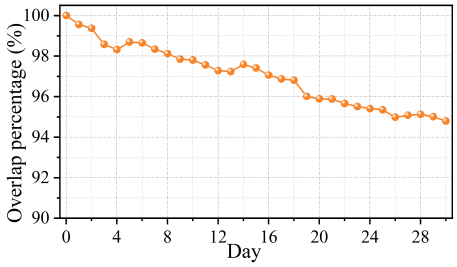


Figure 14: The persistence of initially discovered BSSIDs.

As shown in Figure 14, we calculate the percentage of each day's observed BSSIDs that were present in the initial list—representing the extent to which the initial discovery continues to cover the target BSSIDs over time. We observe a gradual decline in this proportion over time; after one month, 95.4% of the observed BSSIDs still remain from the initial list. This indicates that even if Apple immediately disabled the feature, the attacker's initially discovered BSSIDs would still represent the vast majority of those surrounding the victim's location for a prolonged period.