Artifact Evaluated
NDSS
Available
Functional
Reproduced

NICPrint: Wi-Fi NIC Identification via Malformed Preambles

Cesar Arguello*, Beatrice Perez†, Timothy J. Pierson*, and David Kotz*
*Dartmouth College †University of Massachusetts Boston
carguello.gr@dartmouth.edu, beatrice.perez@umb.edu,
{timothy.j.pierson,david.f.kotz}@dartmouth.edu

***Abstract*—Wi-Fi vulnerabilities are often tied to specific chipsets, yet identifying the chipset behind an observed MAC address from over-the-air transmission alone is an open problem. We present NICPrint, an identification system that recovers the manufacturer and chipset model of a Wi-Fi network interface (NIC) by remotely probing its digital reception logic. The IEEE 802.11 standard specifies what the Wi-Fi preamble must accomplish but not how, leaving vendors free to make implementation choices in their PHY baseband designs. Without requiring network credentials, I/Q access, CSI, or target cooperation beyond ACK transmission, NICPrint elicits these implementation differences by transmitting frames with controlled preamble corruption and observing the polite-ACK responses to construct a fingerprint. We evaluate NICPrint on 25 devices spanning five major Wi-Fi vendors and four Wi-Fi generations. Same-session cross-validation provides an upper bound of 99% accuracy for both manufacturer and chipset identification. On previously unseen devices with chipsets represented in training, NICPrint averages 95% manufacturer and 96% chipset accuracy across the tested classifiers. Under through-wall, non-line-of-sight, and varying distance conditions measured more than one month after training, mean classifier accuracy is 83% for manufacturer and 80% for chipset identification; channel augmentation increases these to 98% and 93%, respectively. These results establish NICPrint as a practical Wi-Fi reconnaissance tool.**

I. Introduction

The wireless landscape comprises billions of devices that communicate across a small handful of standardized protocols, the most pervasive of which is IEEE 802.11 (Wi-Fi) [1]. Although the protocol itself is standardized, the standard allows flexibility in the implementation of network interface cards (NICs). Five vendors – Intel, Qualcomm, Broadcom, MediaTek, and Realtek – account for over 86% of deployed Wi-Fi NICs [2], [3]. Each vendor ships proprietary baseband and firmware implementations of the same standard. Two NICs communicating on the same network may rely on entirely different signal-processing pipelines managing frame detection, carrier frequency offset estimation, and channel equalization. Standard network sniffing does not reveal which implementation a given device uses, particularly with the widespread adoption of MAC-address randomization. This gap between observable network identity and underlying implementation has direct security consequences. An adversary that knows which NIC a device uses may be able to exploit the device. Recent discoveries such as Kr00k [4], Broadpwn [5], FragAttacks [6], and other chipset-specific bugs surfaced by over-the-air Wi-Fi fuzzing [7] are routinely tied to specific vendors or chip families. Identifying a target's chipset lets an adversary select exploits known to affect it rather than blindly attempting attacks that may fail against the target's hardware. Importantly, the aforementioned 86% figure reflects vendor share, not chipset concentration. Each vendor produces dozens of chipset models, making the relevant search space hundreds of chipsets. Brute-forcing this space is not free; trying an exploit against an incompatible chipset may disrupt the target's Wi-Fi stack, force reassociation, or otherwise alert the user. Public databases such as FCC filings can map a known device model to its Wi-Fi chipset, but require the device model as input; an over-the-air observer facing an unknown device (potentially with a randomized MAC address) does not have that information. Furthermore, a defender who can identify the chipsets present on the network can prioritize patching and audit hardware inventories without relying on cooperation from the devices themselves. In both cases, the question is the same: *given a MAC address observed over the air, what vendor and chipset is behind it?*

Prior work on Wi-Fi device fingerprinting addresses related but distinct questions. MAC and driver-layer identification techniques [8]–[10] capture upper-layer artifacts that vendors routinely patch and that reflect driver rather than chipset identity. Clock-skew [11]–[13], radiometric deep-learning [14]–[16], and CSI-based [17], [18] approaches fingerprint individual device instances rather than chipset models, and the latter two additionally require I/Q access or CSI-extraction capabilities, which are unavailable on most receivers. Our proposed system, NICPrint, takes a different route: instead of extracting a fingerprint from analog imperfections of the RF front end, it probes the digital reception logic of the target's PHY baseband.

The key insight in our system is that Wi-Fi NICs differ in how many samples of the preamble they actually use to decode a frame. The IEEE 802.11 standard specifies *what* the Short Training Field (STF) must accomplish but does not mandate *how*. Vendors make design choices that have a direct impact on the number of valid STF samples required for synchronization.

Network and Distributed System Security (NDSS) Symposium 2027
22–26 March 2027, Seoul, Republic of Korea
ISBN 978-1-970672-09-1
https://dx.doi.org/10.14722/ndss.2027.230760
www.ndss-symposium.org

We find that these choices are implementation-specific and rarely modified across product revisions. We demonstrate that by transmitting carefully crafted frames with progressively more corrupted STF samples and monitoring for ACKs, an adversary can measure how a target device's reception degrades as a function of STF corruption. We quantify this measurement in a new metric: Normalized Received ACK Ratio (NRAR) in Section V-C. We use NRAR as input to supervised classifiers that infer the manufacturer and chipset of a target's NIC.

Importantly, an adversary can accomplish NIC identification without requiring the target to be associated with the adversary, share credentials, or run adversary-supplied code. NICPrint leverages the 'polite' Wi-Fi behavior identified by Abedi et al. [19], where a Wi-Fi NIC transmits an ACK in response to any unicast frame addressed to its MAC address, even if the frame comes from outside the network and even when the network is protected by WPA2 or WPA3.

To evaluate NICPrint, we curated a testbed of 25 devices spanning the five major Wi-Fi NIC manufacturers (Intel, Qualcomm, Broadcom, MediaTek, and Realtek) and four Wi-Fi generations (Wi-Fi 4 through Wi-Fi 7). We collected NRAR profiles under static channel conditions and trained supervised classifiers (random forest, logistic regression, and support vector machine) on two tasks: manufacturer identification and chipset identification. To test whether NRAR profiles capture chipset-intrinsic behavior rather than host- or instance-specific artifacts, we tested our system on devices that share chipsets with those in the training set but differ in host hardware (see Section VII). Additionally, we tested the pre-trained models under various channel conditions and with profiles acquired up to six weeks later (see Section VIII). Finally, we enhanced the models trained under static channel conditions with channel augmented data to improve robustness (see Section IX).

In this paper we make the following **contributions**:

- We identify Wi-Fi NIC responses to malformed (non-standard-compliant) preambles as a chipset-discriminative fingerprint and characterize how NRAR profiles, which measure reception degradation under increasing preamble corruption, vary across vendors and chip families.
- We design and implement NICPrint, a chipset and manufacturer identification system of Wi-Fi NICs that leverages the discriminative power of NRAR profiles. NICPrint is implemented using an open-source Wi-Fi stack and requires no network credentials, I/Q access, CSI, or target cooperation beyond ACK transmission.
- We evaluate NICPrint on 25 devices spanning the five major Wi-Fi vendors and four Wi-Fi generations. We also provide evidence that NRAR profiles capture receive-side PHY baseband behavior that persists across different hosts, device instances, and propagation conditions.

II. Wi-Fi Preamble Background

Figure 1 illustrates the structure of an IEEE 802.11 frame for a 20 MHz channel. In this paper, we focus on the 160-sample portion labeled 'Short Training Field' (STF). While we focus on the widely deployed 20 MHz channel, the discussion

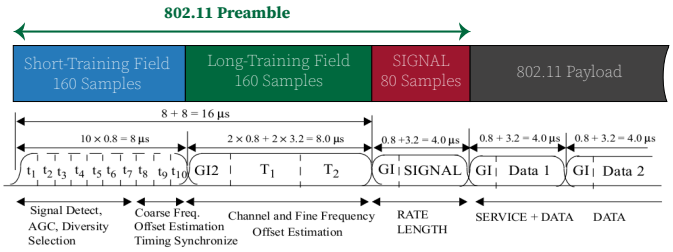


Fig. 1. Structure of a Wi-Fi frame for a 20 MHz channel. Adapted from the IEEE 802.11-2012 standard [20].

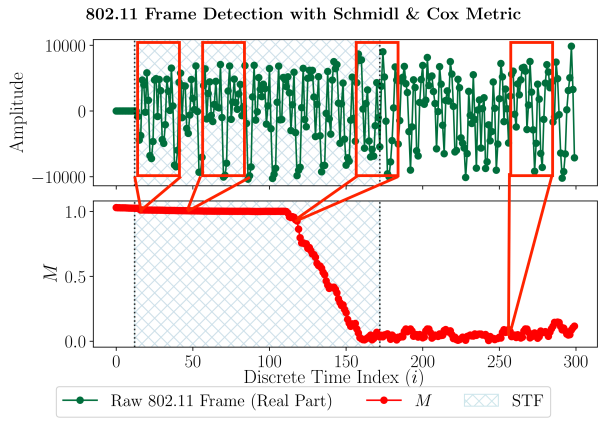


Fig. 2. Wi-Fi frame detection using the Schmidl & Cox algorithm with a correlation window size of 48. The correlation metric M (shown in red) plateaus when computed over samples in the STF and drops sharply once the LTF is encountered.

generalizes directly to wider bandwidth configurations. As shown in Figure 1, the STF consists of ten short repeating symbols defined in the IEEE 802.11 standard [20]. Each short symbol has a duration of 16 samples, resulting in a total STF length of 160 samples. The periodic structure of the STF enables two primary functions in frame decoding: frame detection and coarse carrier frequency offset (CFO) estimation. A brief discussion of each follows.

Frame detection. A widely used method for frame detection in Wi-Fi is the Schmidl & Cox algorithm [21]. The core idea is to exploit the periodicity of the STF by computing a normalized autocorrelation metric $M[i]$ that attains high values when the repeated structure of the STF is present and significantly lower otherwise, as shown in Figure 2. Notably, $M[i]$ plateaus when the correlation window lies entirely within the STF. A receiver declares frame detection when the metric exceeds a threshold for a sufficient number of consecutive samples, a system parameter we refer to as the plateau length. As a result, frame detection depends only on a subset of STF samples whose size is determined by the plateau length and the correlation window size.

Coarse CFO estimation. Mismatches between the crystal oscillators of a transmitter and receiver introduce CFO, a time-varying phase rotation that degrades decoding if left uncorrected. The STF enables a coarse CFO estimate, which

is later refined using the Long Training Field (LTF) and pilots. Since samples that are 16 time indices apart in the STF are identical at the transmitter, any observed phase difference corresponds to phase accumulated over 16 samples due to CFO. The receiver compensates by applying the inverse rotation. As with frame detection, practical NICs use only a subset of STF samples for CFO estimation.

III. Related work

Identifying wireless devices from their transmissions has been studied for more than two decades. Existing identification techniques tend to fall into one of four categories: (1) MAC and driver-layer fingerprinting, (2) clock-skew and timing-based fingerprinting, (3) radiometric and RF-impairment fingerprinting, and (4) preamble-layer manipulation. Table I summarizes a representative sample of the existing techniques and compares them with our approach. Note that the table omits preamble-layer manipulation techniques, since they either target different protocols or have objectives that are not directly comparable to NICPrint's objectives. A brief discussion for each of the aforementioned categories follows.

MAC and driver-layer fingerprinting. These techniques exploit the heterogeneity in vendor implementations of the IEEE 802.11 MAC layer and driver software. This category is further divided into two subcategories tackling different problems: per-implementation fingerprinting and per-instance fingerprinting. The former attempts to identify the driver, device model, or vendor behind a frame, while the latter links frames from the same device across transmissions.

Among per-implementation efforts, Franklin et al. [8] showed that probe-request inter-frame timing produces a recognizable driver signature. Gopinath et al. [23] demonstrated that implementation differences in backoff timers, virtual carrier sensing (NAV), and other CSMA/CA parameters can be used to fingerprint device vendors. Bratus et al. [22] developed an active approach for driver fingerprinting where non-compliant 802.11 frames (e.g., frame types with forbidden flag values) are transmitted and the responses (or lack thereof) recorded to create a fingerprint. These techniques, however, fingerprint software the vendor can revise. NICPrint instead probes PHY baseband behavior that is rarely altered across product revisions.

Among per-instance work, Neumann et al. [10] showed that passive observation of frame metadata (e.g., rate selection, frame sizes, frame inter-arrival time) is sufficient to discriminate among 802.11 device instances. Robyns et al. [9] showed that a per-bit entropy analysis of a single frame yields a fingerprint that is 80% unique among 50 devices, decaying to 15% at 10,000 devices. Other work re-links devices across MAC randomization using probe-request entropy [24], inter-frame timing [29], or MAC layer implementation flaws [30]; these techniques determine whether a transmission originates from a previously seen device instance without revealing its chipset. NICPrint does not aim to re-link transmissions to a specific device instance but to identify the manufacturer and chipset of the NIC that produced them.

Clock-skew and timing-based fingerprinting. Kohno et al. introduced device fingerprinting based on clock skew estimates from TCP and ICMP timestamps, showing that deviations in the system's clock frequency yield per-device signatures [11]. Jana and Kasera adapted this idea to 802.11, obtaining more accurate estimates by leveraging beacon timestamps at the MAC layer [12]. Other work has applied clock-skew fingerprinting to access-point detection, virtual-machine identification, and IoT device classification [13], [25], [31]. Although clock-skew-based fingerprinting methods are passive and low-cost, their reliability degrades under temperature variation and deployment heterogeneity [25], [26], [32]. More importantly, these methods aim to distinguish individual device instances while NICPrint identifies the chipset and manufacturer of a device's NIC, which directly informs attacks targeting specific Wi-Fi stacks.

Radiometric and RF-impairment fingerprinting. Recent surveys cover this body of work comprehensively [33], [34]. Similar to clock-skew approaches, methods in this category fingerprint individual device instances. Note that chipset identification is a goal distinct from instance identification. RF-based fingerprinting performs instance identification by determining whether two transmissions originate from the same radio. Since the characteristics exploited by these methods (e.g., CFO, I/Q imbalance, and amplifier nonlinearities) reflect manufacturing variations across nominally identical devices, they cannot reveal the chipset used by said device. NICPrint, on the other hand, identifies the chipset regardless of which device instance generated a transmission. Moreover, the fingerprints these techniques extract degrade rapidly over time. Recent work shows that classifier accuracy halves when training and testing on different days due to channel variability, hardware warm-up, and power-cycling [27], [28]. NICPrint's fingerprint, by contrast, remains stable across time and under various channel conditions. Finally, many of these techniques depend on raw I/Q samples at the receiver; CSI-based variants [17], [18] relax the I/Q requirement but still depend on CSI-extraction tools. NICPrint requires no I/Q or CSI access on the receive side.

Preamble-layer manipulation. Finally, the fourth category examines techniques that manipulate the preamble itself. Jenkins et al. showed that varying the length and symbol content of the IEEE 802.15.4 preamble can fingerprint chipsets used in wireless sensor networks [35]. NICPrint builds on this idea for IEEE 802.11, whose more elaborate preamble structure allows probing the NIC behavior at the sample level rather than by altering discrete preamble symbols.

Bloessl et al. showed that the 802.11p scrambler seed is predictable and can be used to track vehicular radios across MAC randomization [36]. Vanhoef et al. later confirmed that commodity Wi-Fi radios exhibit similarly predictable seed behavior [24]. Preamble Injection and Spoofing (PrInS) attacks [37] exploit the lack of authentication on the preamble to silence the channel or force frame deferral. Unlike these latter works, which use the preamble for tracking or disruption, NICPrint uses controlled preamble corruption as a NIC probe.

TABLE I
COMPARISON OF NICPRINT WITH REPRESENTATIVE WIRELESS FINGERPRINTING APPROACHES.

Approach	Target	Required traffic	Hardware requirements	Cross-target transfer	Temporal stability
MAC/driver layer					
Franklin et al. [8]	Driver	Probe requests	Monitor-mode NIC	✓	High†
Bratus et al. [22]	Driver	Injected-frame responses	Monitor-mode NIC with injection	✓	High†
Gopinath et al. [23]	Host manufacturer	Management frames	Monitor-mode NIC	✓	High†
Vanhoef et al. [24]	Instance + host	Probe requests	Monitor-mode NIC	✓	High†
Neumann et al. [10]	Instance	Any frame type	Monitor-mode NIC	✗	Unknown
Clock/timing					
Clock skew [12], [13]	Instance	Beacons, probe responses	Monitor-mode NIC	✗	Low [25], [26]
GTID [25]	Instance + host	Any frame type	Monitor-mode NIC	✓	Unknown
RF-based					
Radiometric [14]–[16]	Instance	Any frame type	I/Q receiver	✗	Low–Med. [27], [28]
CSI [17], [18]	Instance	Any frame type	CSI-capable NIC or SDR	✗	High [17]
This work					
NICPrint	**NIC chipset + manufacturer**	**Elicited ACKs**	**Custom PHY NIC (e.g., AntSDR)**	✓	**High**

Target: *host* = a device model (e.g., a MacBook Pro), *instance* = a device instance (e.g., a particular MacBook Pro).
Cross-target transfer: ✓ = fingerprint generalizes to unseen targets of the same class, ✗ = fingerprint is specific to measured targets.
Temporal stability: drift of the target's fingerprint across days to weeks, with no retraining between collection and use.
† MAC- and driver-layer behavior derived from implementation choices is inherently stable over time, but can be patched by vendors or users.

IV. SECURITY MODEL

In this section, we describe the assumed capabilities of the adversary that uses NICPrint and the objectives that motivate its use. We focus on a victim environment consisting of one or more devices equipped with Wi-Fi NICs, where the underlying network may be open or protected by WPA2 or WPA3. Hereafter, we use the term *target* to refer to any Wi-Fi device whose NIC the adversary seeks to identify.

A. Adversary Goals

The adversary seeks to perform *wireless reconnaissance for targeted exploitation*. Their objective is to identify the chipset and manufacturer of each device's Wi-Fi NIC in a physical environment, so that subsequent attacks can be directed at devices known to be vulnerable to chipset-specific exploits. Note that the adversary seeks to determine whether a vulnerable chipset is present in range, not to link a MAC address to a specific device or user; that linkage is orthogonal and addressed by the existing derandomization literature [24], [29], [30]. We therefore define the adversary's goals as manufacturer and chipset identification.

Manufacturer Identification. For each target device, identify the manufacturer of its Wi-Fi NIC (e.g., Broadcom, Intel, MediaTek, Qualcomm, Realtek). Manufacturer identification is operationally valuable when the adversary has an exploit that applies broadly across a vendor's product line. Additionally, vendor identification supports more accurate chipset identification, as discussed in Section VIII.

Chipset Identification. For each target device, identify its specific Wi-Fi chipset (e.g., `BCM4360`, `MT7921`, `AX210`). Chipset identification enables the adversary to focus subsequent exploitation on chipsets known to be vulnerable. It also constrains the device-model search space since some Wi-Fi chipsets appear only in specific consumer products; thus, permitting device identification and tracking in some cases.

B. Threat Model

The adversary is modeled as an individual either physically present within radio range of the target environment or remotely controlling a device so positioned. We elaborate on the adversary's capabilities and actions next.

Capabilities. The adversary is equipped with two radios. The first is a commodity Wi-Fi NIC operating in monitor mode, used to passively capture 802.11 frames. The second is a software-defined radio (SDR) capable of transmitting both standards-compliant Wi-Fi frames and frames with controlled deviations from the 802.11 specification; such SDRs are commercially available at moderate cost. Both radios operate within the regulatory bands and channels supported by commodity Wi-Fi hardware. Additionally, the adversary has access to a reference set of Wi-Fi NICs containing the chipsets expected in target environments. The reference set is necessary to create labeled training data during the profile phase discussed below.

Actions. The adversary engages in three phases: profile, enumerate, and active probing. In the *profile phase*, the adversary uses its own collection of common Wi-Fi NICs to measure their NRAR and trains one or more classifiers to identify them. Next, in the *enumeration phase*, the adversary identifies target MAC addresses in the target environment, either by listening on Wi-Fi channels of interest and harvesting addresses from observable frames (beacons, probe requests, ACKs) or from prior knowledge. Finally, in the *active probing phase*, the adversary transmits crafted frames to selected target MAC addresses and observes the resulting responses. The adversary then classifies those responses using the models developed in the profile phase.

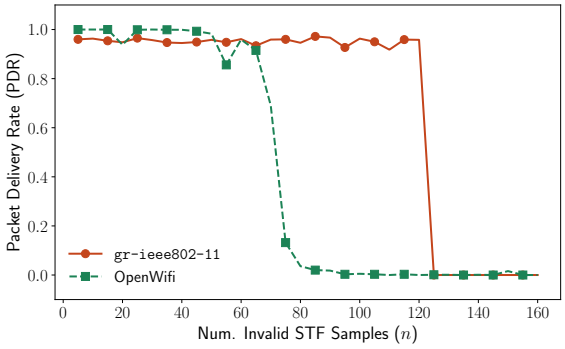


Fig. 3. Packet Delivery Rate (PDR) for frames with different numbers of invalid STF samples for a USRP N210 running `gr-ieee802-11` and an AntSDR E200 running OpenWiFi. PDR decreases sharply after the number of *invalid* samples is 160 minus the minimum requirement for the target platform.

V. NICPrint

In this section, we describe how malformed preambles form the basis of a fingerprint for Wi-Fi NICs and describe our system, NICPrint, in detail.

A. Malformed Preambles as Fingerprints

The IEEE 802.11 standard specifies the structure of the STF and the functions it must provide. It does not mandate specific methods or algorithms to implement the frame detection or coarse CFO estimation functions. Hence, Wi-Fi implementers are free to design proprietary processing pipelines. Open-source implementations of the Wi-Fi protocol, such as Bloessl's `gr-ieee802-11` [38] and OpenWiFi [39], differ in key system parameters such as plateau length. As a result, even though the Wi-Fi standard calls for 160 STF samples, OpenWiFi requires approximately 100 STF samples for successful synchronization, while `gr-ieee802-11` requires only around 50. We validated this observation by transmitting frames with progressively fewer valid STF samples (i.e., setting n contiguous STF samples to *zero* amplitude) and measuring the packet delivery rate (PDR), defined as the fraction of successfully received frames, in an SDR implementing the target receiver, as shown in Figure 3. As expected, PDR decreases sharply after the number of valid STF samples falls below the implementation-specific requirement (i.e., after the number of *invalid* samples is 160 minus the minimum requirement above). This behavior arises because frame detection fails when the last correlation window does not contain sufficient valid STF samples to satisfy the required plateau length. Even when detection succeeds, an insufficient number of valid STF samples degrades the CFO estimate, leading to poor channel estimation and bit decoding. This finding across two different open-source implementations suggests that commercial off-the-shelf (COTS) Wi-Fi NICs may also exhibit similar behavior.

B. Testbed

We curated a set of 25 devices equipped with Wi-Fi NICs from the five major manufacturers covering 86% of the market. For each manufacturer, we selected five devices spanning multiple Wi-Fi generations, including generations 4, 5, 6, and 7. Together these cards represent the standards supported by the vast majority of deployed Wi-Fi NICs to date [3]. The selected devices include a mix of laptops, embedded platforms, and development boards, providing hardware diversity. The specific devices are listed in Table II.

TABLE II
Wi-Fi Chipset Device Reference List

Device Name	Wi-Fi Chip	Manufacturer	Standard
RPi 4	BCM4345C0		Wi-Fi 5
MacBook Pro 2012	BCM4331		Wi-Fi 4
Motorola Nexus 6	BCM4356	Broadcom	Wi-Fi 5
MacBook Pro 2013	BCM4360		Wi-Fi 5
MacBook Pro 2019	BCM4364		Wi-Fi 5
TP-Link AC600	RTL8821AU		Wi-Fi 5
Linksys WUSB6300	RTL8812AU		Wi-Fi 5
Fenvi PCIe Board	RTL8852CE	Realtek	Wi-Fi 6
Fenvi PCIe Board	RTL8822CE		Wi-Fi 5
AWUS1900	RTL8814AU		Wi-Fi 5
AWUS036ACM	MT7612U		Wi-Fi 5
Motorola G Power	MT6765		Wi-Fi 5
AWUS036AXML	MT7961	MediaTek	Wi-Fi 6
MT7925 (PCIe)	MT7925		Wi-Fi 7
MT7921 (PCIe)	MT7921		Wi-Fi 6
Asus ROG	AC9560		Wi-Fi 5
Lenovo Yoga i7	AX201		Wi-Fi 6
Dell Latitude	AC8265	Intel	Wi-Fi 5
Intel AX210 (M.2)	AX210		Wi-Fi 6
Intel AC3160 (M.2)	AC3160		Wi-Fi 5
QCA9377 (M.2)	QCA9377		Wi-Fi 5
QCA6174 (M.2)	QCA6174		Wi-Fi 5
Netgear Orbi	QCA9984	Qualcomm	Wi-Fi 5
QCNCM865 (PCIe)	WCN7850		Wi-Fi 7
Netgear Orbi	IPQ4019		Wi-Fi 5

C. Normalized Received ACK Ratio (NRAR)

Measuring the response of COTS Wi-Fi NICs to malformed preambles (preambles with n samples set to zero amplitude) via packet delivery rate, as in Section V-A, is not always feasible in practice. NICs without support for 'monitor mode' complicate or prevent PHY-layer PDR measurements; indeed, most devices in our testbed lack this capability. This constraint made it impossible to precisely count the number of malformed frames successfully decoded.

To address this limitation, we leverage the 'polite' behavior of Wi-Fi identified by Abedi et al. [19]. A Wi-Fi NIC automatically transmits an IEEE 802.11 ACK frame whenever a received frame (i) is addressed to its MAC address, (ii) has a valid MAC header corresponding to a data frame, and (iii) passes the frame check sequence (FCS). Crucially, this behavior persists even in protected networks (e.g., WPA2/WPA3). Although data frames are encrypted and a NIC should theoretically only respond to frames encrypted with the correct key, to meet stringent short interframe space timing constraints set by the 802.11 specification, ACK generation occurs before decryption and sender authentication are completed. As a result, the NIC transmits an ACK for any frame that satisfies the

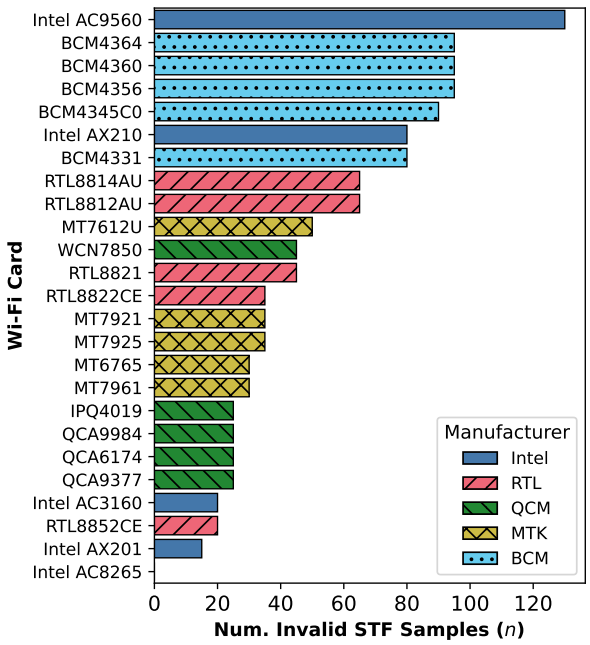


Fig. 4. Number of invalid STF samples n required to decrease NRAR below 95% for all Wi-Fi cards in our testbed.

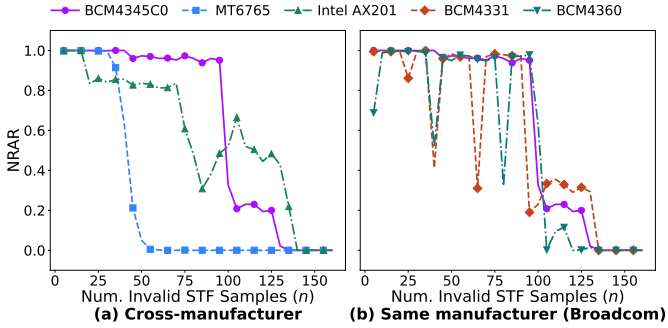


Fig. 5. (a) NRAR profiles for three devices from different manufacturers. (b) NRAR profiles for three Broadcom chips.

above conditions, regardless of whether it originates from a legitimate sender. We observed that every Wi-Fi NIC on the testbed above complies with this 'polite' behavior. Moreover, the malformed frames that elicit polite ACKs are silently discarded by the target NIC without surfacing to the host operating system. Therefore, host-based intrusion-detection systems do not observe any probing activity. Although Wi-Fi-aware network intrusion-detection systems could, in principle, detect illegitimate transmissions by monitoring all wireless transmissions in a channel, their deployment is rare in practice.

By transmitting frames with malformed preambles and monitoring for ACK responses, we can thus estimate frame reception at the target. While we cannot directly observe how many malformed frames the target decodes, counting the returned ACKs provides a practical proxy for reception. Repeating this process for different values of n yields the received ACK ratio (RAR), defined as the ratio between the number of received ACKs and the number of transmitted frames. RAR provides an indirect estimate of PDR, since frames that are not successfully received (e.g., due to PHY decoding errors or FCS failure) are not ACKed by the target NIC. RAR, however, does not perfectly track PDR, as ACK frames themselves may be lost due to channel impairments.

To mitigate the dependency on channel impairments, we define the normalized received ACK ratio (NRAR) as:

$$\mathrm{NRAR}(n) = \frac{\mathrm{RAR}(n)}{\mathrm{RAR}(0)}$$

where $\mathrm{RAR}(0)$ corresponds to the received ACK ratio obtained using a standard-compliant preamble (i.e., $n = 0$).

Figure 4 shows the number of invalid STF samples n required for NRAR to decrease below 95% for all Wi-Fi cards in our testbed. Since a 95% PDR is commonly considered reliable in prior work [40], we define the minimum number of invalid STF samples required to reduce NRAR below this threshold as n_{min}. In Figure 4 we observe that n_{min} often clusters across NICs from the same manufacturer (indicated by the color and pattern of each bar), highlighting the potential of NRAR-derived features (e.g., n_{min}) for NIC identification. Note that most NICs from a given manufacturer exhibit almost the same n_{min} value, likely due to reuse of PHY-layer designs.

Throughout this paper, we refer to the measured function $\mathrm{NRAR}(n)$ as the *NRAR profile*. Given the discrete structure of the STF, n can take integer values in the range $0 < n \leq 160$. Figure 5a shows the NRAR profile for three devices from different manufacturers in our testbed. For each device, n_{min} is sufficient to distinguish it from the others. Moreover, the overall NRAR profile varies significantly across NICs. For instance, in the `MT6765`, frame reception drops sharply to zero once approximately 30 STF samples are corrupted. In contrast, the `BCM4345C0` and the `AX201` exhibit a more gradual, step-like degradation as n increases. Smaller intra-vendor differences also appear in the full profile (see Figure 5b). For example, the `BCM4360` degrades to zero almost immediately at n_{min}, while the `BCM4331` and `BCM4345C0` exhibit step-like plateaus at different NRAR levels before fully degrading to zero. Both n_{min} and the full profile shape are therefore implementation-specific. We use the full NRAR profile as our fingerprint.

D. System Design & Implementation

Figure 6a illustrates the architecture of NICPrint. The system comprises four components: the frame injector, ACK receptor, data collection controller, and classifier module. We describe each component below.

Frame Injector. The frame injector is a Wi-Fi NIC capable of transmitting frames with a controlled number of invalid STF samples (i.e., malformed preambles). In our implementation, we used an AntSDR E200 running a modified version of OpenWiFi. We modified both the PHY baseband and driver of OpenWiFi to support fine-grained STF manipulation.[1] Specifi-

[1]Our fork of OpenWiFi is available in the paper's artifact [41].

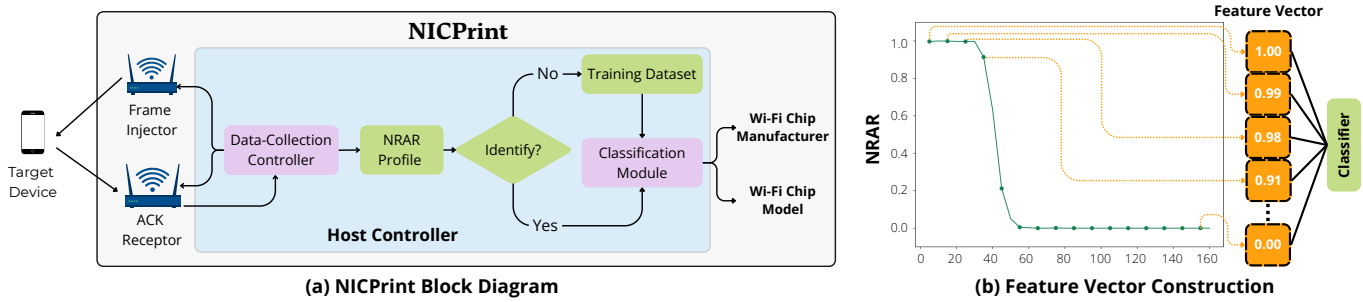


Fig. 6. NICPrint architecture: (a) Block diagram with all system components, (b) Example of how the input to the classifier is constructed.

cally, our implementation zeroed out the last n contiguous samples of the STF for each transmitted frame. Because the AntSDR E200 runs an embedded Linux stack, frame construction was handled at the injector. The data-collection controller provided configuration parameters – including the target MAC address, transmitter MAC fields, and transmission triggers. While a specific transmitter MAC address is not strictly required to elicit ACKs, assigning known addresses allowed us to associate received ACKs with transmitted frames. Our implementation transmitted all frames using BPSK-1/2. Although NRAR profiling can be performed with any modulation scheme, using the lowest-rate scheme improves reception reliability, as discussed in Section XI.

ACK Receptor. The ACK receptor is a Wi-Fi NIC operating in monitor mode. We used an Alfa Networks AWUS036ACM in our implementation. The receptor passively monitored the channel for IEEE 802.11 ACK frames following each transmission. Since ACK frames do not carry a source address, they were associated with transmitted frames based on the known MAC address of the injector, which shows as the destination address in the ACK frames, set by the data-collection controller. Captured ACKs were then forwarded to the data-collection controller, which used them to compute the NRAR profile as described in Section V-C. Although injection and reception could be co-located on a single device, we used separate NICs due to the limited receiver sensitivity of the AntSDR E200 in the 5 GHz band. A higher-sensitivity device would enable a single-NIC implementation.

Data-Collection Controller. The data-collection controller is a Linux-based machine that orchestrates the identification process. It configures system parameters for both the frame injector and ACK receptor. These parameters include the number of invalid STF samples n, the operating channel, and MAC addresses used for ACK discrimination. This component also synchronizes transmission and reception, in addition to driving the construction of NRAR profiles. In our implementation, the controller generated an NRAR profile by first setting a transmission of p frames with $n = 0$ to obtain a baseline RAR; it then iterated n from 0 to 160 in steps of s, setting a transmission of p frames at each step and computing $\mathrm{NRAR}(n)$ as the ratio of step RAR to baseline RAR. We used $p = 1000$ and $s = 5$ as our default settings, but discuss parameter selection in Section XI. Using the AntSDR E200's fastest reliable transmission rate of 1000 frames per second, a full profile measurement took around 33 seconds with the aforementioned default parameters.

Classifier Module. The classifier module operates on the NRAR profiles extracted for each target NIC. As discussed in Section V-C, NRAR profiles exhibit both coarse- and fine-grained variations across devices which capture how frame reception degrades as STF corruption increases. These differences manifest in characteristics such as the degradation rate beyond n_{min} and the presence of step-like transitions, which reflect implementation-specific PHY-layer behavior. Since these characteristics are inherently encoded in the NRAR profile, we use its measured NRAR values directly, without additional feature engineering. To construct the classifier input, we represented each NRAR profile as a feature vector by arranging the measured $\mathrm{NRAR}(n)$ values in increasing order of n, as illustrated in Figure 6b. For example, the profile shown in Figure 6b produces a feature vector of the form $[1.00, 0.99, 0.98, 0.91, \ldots, 0.00]$, where each entry corresponds to the NRAR measured at a successive value of n. To evaluate the discriminative power of NRAR profiles, we employed standard supervised-learning models, including random forests (RF), logistic regression (LR), and support vector machines (SVM), implemented using the `scikit-learn` library [42]. For each model family, we trained two separate classifiers from the same NRAR profile dataset: one that infers the manufacturer of the target NIC and one that infers its chipset. The classifier module handles model training, which constitutes the first step of the identification process (see Section V-E). We tuned the hyperparameters for each model using `GridSearchCV` [43] over a limited set of candidate values. Note that `GridSearchCV` tuned via a 3-fold split within each training set, so held-out evaluation sets are never involved in parameter selection. Although we used RF, SVM, and LR in our implementation, NICPrint is classifier-agnostic and could incorporate alternative models.

E. NIC Identification

As mentioned in Section IV, Wi-Fi NIC identification is a three-phase process, described below.

Profile Phase. In this phase, the adversary curates a dataset of NRAR profiles for Wi-Fi NICs of interest. We assume the

adversary has access to a variety of devices equipped with common NICs of interest. For each NIC, the adversary generates multiple NRAR profiles using the procedure described in Section V-D, which are then used to train a manufacturer and chipset classifier. To ensure reliable measurements, the target device is placed at a short distance d (e.g., 1 meter) from the frame injector and ACK receptor in a line-of-sight (LOS) configuration.[2] This arrangement increases the likelihood that injected frames are successfully received by the target device and that the resulting ACKs are captured by the receptor. Collecting multiple profiles per device enables the classifier to capture variability in NRAR measurements and learn NIC-specific characteristics. Note that the number of profiles chosen throughout the paper differ by role; the same-session evaluation described in Section VI uses 100 profiles per device so that classifiers can learn each chipset's within-class variability; the evaluations in later sections use 30 profiles per device, since they only test pre-trained models.

Enumerate Phase. The adversary harvests target MAC addresses in the target environment using standard channel-monitoring tools, or prior knowledge. The output of this phase is a list of candidate MAC addresses to probe.

Active Probing Phase. The adversary deploys NICPrint in the target environment with the classifiers trained during the profile phase. For each MAC address identified during the enumerate phase, the adversary collects a NRAR profile and runs it through the trained classifier, which outputs the inferred Wi-Fi NIC's manufacturer and chipset. The adversary can then use this information to attack devices with specific Wi-Fi NICs by exploiting vulnerabilities associated with those NICs.

VI. Manufacturer and Wi-Fi Chip Classification

We evaluated NICPrint's ability to fingerprint Wi-Fi NICs by curating a dataset of NRAR profiles for the 25 devices in the testbed introduced in Section V-B. Each device was associated with a WPA2-protected 802.11 network under our control on Wi-Fi channel 165 (5.825 GHz).[3] We collected 100 profiles per device (following the 1m LOS procedure described in Section V) in a laboratory inside a busy university building with typical background traffic, for a final dataset of 2,500 NRAR profiles. We trained the supervised classifiers described in Section V-D and evaluated them using stratified five-fold cross-validation. Stratification ensures that each fold preserves the class distribution, which we intentionally balance. Each NRAR profile constitutes an independent sample, and each sample appeared in the training split for four folds and in the validation split for one fold. We report mean classification accuracy across folds as our primary metric, but also report per-class precision, recall, and F1-score for completeness. Note that this evaluation trains and tests on the same devices and therefore establishes an upper bound on NRAR's discriminative power. Section VII evaluates whether this discriminative power is transferred to previously unseen devices, and Section VIII evaluates whether discrimination sustains large temporal gaps and different channel conditions. In Section VIII, we also explore a unified two-stage classifier in which chipset classification depends on the inferred manufacturer.

A. Manufacturer Identification

All classifiers achieved an average accuracy of 99% across the five folds. Table III reports per-class precision, recall, and F1-score. Note that throughout the paper, we report uncertainties in tables and figures as 95% confidence intervals (CIs) computed with the bootstrap method [44], [45]. Figure 15 in Appendix A shows the confusion matrices for each classifier. These results show that NRAR profiles exhibit strong manufacturer-specific structure, enabling near-perfect discrimination across vendors. Although manufacturer identification provides only coarse-grained information about a target device, it plays a critical role in practical NIC identification. In particular, as we show in Section VIII, manufacturer classification remains robust under challenging channel conditions (e.g., longer distances or partial obstruction), where fine-grained chipset classifiers degrade significantly.

TABLE III
Per-class precision (P), recall (R), and F1-score (F1) for the manufacturer identification task.

	RF			SVM			LR		
Class	P	R	F1	P	R	F1	P	R	F1
BCM	1.00	1.00	1.00	0.99	0.98	0.99	1.00	0.96	0.98
RTL	0.99	1.00	1.00	0.99	1.00	1.00	0.98	1.00	0.99
MTK	1.00	0.99	1.00	1.00	0.99	0.99	0.99	0.99	0.99
Intel	1.00	1.00	1.00	0.99	1.00	0.99	0.98	1.00	0.99
QCM	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99
Avg	**1.00**	**1.00**	**1.00**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**

Uncertainties: no 95% CI half-width exceeds 0.02, and the median half-width is at most 0.01 for every classifier.
Class names: Broadcom (BCM), Realtek (RTL), MediaTek (MTK), and Qualcomm (QCM).

B. Wi-Fi Chipset Identification

RF, SVM, and LR classifiers achieved average accuracies of 99%, 96%, and 95%, respectively, across all cross-validation folds. Table IV reports per-class precision, recall, and F1-scores for each classifier. As expected, chipset classification yields lower accuracy than manufacturer classification. This happens due to the reduced separability between Wi-Fi NICs from the same manufacturer, as discussed in Section V-C. The majority of misclassifications in this evaluation occurred between chipsets from the same manufacturer. In particular, 86% of all misclassified samples corresponded to confusion between chipsets from the same vendor, reinforcing the observation that intra-manufacturer distinctions are more subtle than inter-manufacturer ones. This trend is particularly pronounced for Broadcom and Qualcomm chips. An extended discussion on the per-class failures is deferred to Appendix A. Nevertheless,

[2]For simplicity, we assume the frame injector and ACK receptor are co-located, as in all experiments in this paper.

[3]We conducted experiments with the Qualcomm `IPQ4019` on channel 64, as it does not support channel 165.

TABLE IV
PER-CHIPSET PRECISION (P), RECALL (R), AND F1-SCORE FOR THE CHIPSET IDENTIFICATION TASK.

	RF			SVM			LR		
Chipset	P	R	F1	P	R	F1	P	R	F1
BCM4345C0	1.00	1.00	1.00	0.98	0.98	0.98	0.97	0.97	0.97
BCM4331	1.00	1.00	1.00	0.99	0.84	0.91	0.97	0.89	0.93
BCM4356	0.97	0.98	0.97	0.79	0.89	0.84	0.78	0.86	0.82
BCM4360	1.00	0.98	0.99	0.91	0.84	0.88	0.93	0.79	0.85
BCM4364	1.00	1.00	1.00	1.00	1.00	1.00	0.95	1.00	0.98
RTL8821AU	0.99	1.00	0.99	1.00	1.00	1.00	0.99	1.00	0.99
RTL8812AU	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RTL8852CE	1.00	1.00	1.00	0.98	1.00	0.99	0.97	1.00	0.98
RTL8822CE	1.00	1.00	1.00	0.99	1.00	0.99	1.00	0.98	0.99
RTL8814AU	1.00	1.00	1.00	0.98	1.00	0.99	1.00	1.00	1.00
MT7961	1.00	0.97	0.98	0.97	0.93	0.95	0.94	0.90	0.92
MT7612U	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
MT6765	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.99	0.98
MT7925	0.99	1.00	0.99	0.99	0.98	0.98	0.93	0.97	0.95
MT7921	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.99	0.98
AC9560	1.00	1.00	1.00	0.99	1.00	0.99	1.00	1.00	1.00
AX201	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
AC8265	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
AX210	1.00	1.00	1.00	0.94	1.00	0.97	0.99	1.00	0.99
AC3160	1.00	1.00	1.00	0.99	1.00	0.99	0.99	1.00	0.99
QCA9377	0.90	1.00	0.95	0.76	0.65	0.70	0.70	0.69	0.69
QCA6174	1.00	0.90	0.95	0.69	0.79	0.74	0.69	0.70	0.70
QCA9984	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00	0.99
WCN7850	1.00	1.00	1.00	0.99	1.00	0.99	1.00	1.00	1.00
IPQ4019	0.99	1.00	0.99	0.99	1.00	0.99	0.98	0.98	0.98
Avg	**0.99**	**0.99**	**0.99**	**0.96**	**0.96**	**0.96**	**0.95**	**0.95**	**0.95**

Uncertainties: median half-widths (95% CI) are 0.00/0.01/0.02 for RF/SVM/LR, with maxima of 0.06/0.10/0.09, all attributable to QCA9377 and QCA6174.

chipset classifiers performed well across all validation folds, indicating that NRAR profiles provide reliable, though not perfect, separability for chipset identification.

VII. PREVIOUSLY UNSEEN DEVICES

Section VI establishes that NRAR profiles are a viable fingerprint primitive. Classifiers trained and tested on the same set of devices identified both manufacturer and chipset of their Wi-Fi NICs with high accuracy. That evaluation, however, did not test a core assumption on which NICPrint relies: NRAR profiles capture chipset-intrinsic PHY behavior rather than host- or instance-specific artifacts. A classifier trained on a single device per chipset could have learned artifacts, product of antenna geometry or RF front-end characteristics, that happen to correlate with chipset identity in the training set. We tested whether NRAR profiles capture chipset-intrinsic behavior by conducting a cross-device validation using devices that were *not* part of the training set but whose Wi-Fi chipsets are represented in the testbed of Section V-B. In total, the cross-device dataset comprised five devices spanning all five manufacturers in our testbed. The new devices fall into two categories: *different host platform* and *different physical instance*. In the different host platform case, a testbed chipset appears in a different product, such as an iMac sharing the `BCM4360` with the MacBook Pro in the testbed. In the different physical instance case, the device is a second unit of a host model in the testbed, such as a second AWUS036AXML. Table V labels each device

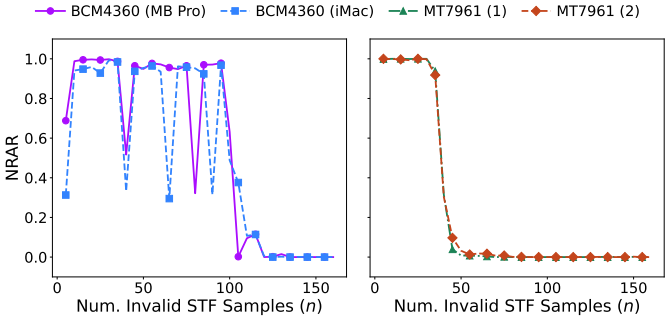


Fig. 7. NRAR profiles of two Wi-Fi NICs from Table V overlaid with NRAR profiles of their counterparts in the original testbed (Table II).

accordingly. Both categories test the same property: whether the classifiers identify the manufacturer and chipset of the NIC in devices an adversary does not control.

TABLE V
DEVICES USED FOR CROSS-DEVICE EVALUATION.

Device	Wi-Fi Chip	Manufacturer	Difference
iMac 2014	BCM4360	BCM	Host Platform
BrosTrend AC650	RTL8821AU	RTL	Host Platform
AWUS036AXML	MT7961	MTK	Physical Instance
Dell Latitude	AC8265	Intel	Physical Instance
QCA9377 (M.2)	QCA9377	QCM	Physical Instance

We collected 30 NRAR profiles per device in Table V, for a total of 150 new NRAR profiles, using the procedure described in Section V, in the same laboratory environment and setup as the training dataset from Section VI, but at least one week after the training measurements were taken. These new profiles are only used for testing pre-trained classifiers since they represent profiles that would be acquired during the *active probing* phase (i.e., these are profiles from devices not under the adversary's control). We trained the supervised classifiers described in Section V-D exclusively on the 25-device testbed of Section V-B (i.e., the 2,500 NRAR profiles from Section VI); no profiles from the devices in Table V appeared in training. Note that the one-week collection gap also introduces incidental variation in channel conditions; a systematic evaluation of NICPrint's robustness to varying wireless channel conditions, however, is deferred to Section VIII.

Figure 7 shows that NRAR profiles from different devices that share the same Wi-Fi NIC are nearly identical for both categories (different host platform and different physical instance). The traces for the two `MT7961` instances (different physical instance case) overlap almost exactly, while the two `BCM4360` traces (different host platform case) show small differences in the "dips" of the profile. Since the locations of the "dips" vary across repeated measurements of the same device, we hypothesize that they are measurement artifacts rather than chipset-intrinsic behavior. These "dips" are likely the result of bursty channel contention or aggressive power-saving behavior in the target NIC. More importantly, because the dip positions are not stable across measurements, the classifiers treat them

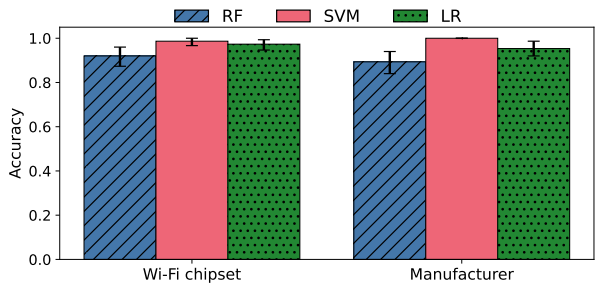


Fig. 8. Accuracy for the cross-device evaluation. Classifiers are trained exclusively on NRAR profiles from devices in Table II and evaluated on NRAR profiles from the previously unseen devices in Table V.

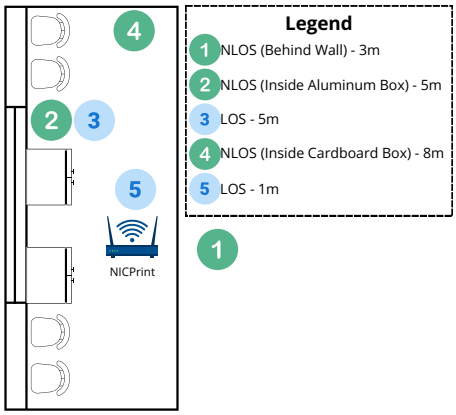


Fig. 9. Floor plan of our laboratory with each test position relative to NICPrint's frame injector. Positions 1, 2, and 4 are non-line-of-sight (NLOS); Positions 3 and 5 are line-of-sight (LOS). Position 5 replicates the channel conditions of the training data collection and serves as a temporal-stability control.

as noise rather than as discriminative features. Beside these dips, the BCM4360 in the iMac follows the same pattern as its counterpart in the MacBook Pro.

While Figure 7 provides visual evidence that NRAR profiles generalize across different hosts and instances, Figure 8 confirms that classifiers trained on NRAR profiles from one set of devices (i.e., those under the adversary's control) identify the manufacturer and chipset of previously unseen devices with high accuracy. RF, SVM, and LR achieved chipset-classification accuracies of 92%, 99%, and 97%, respectively, and manufacturer-classification accuracies of 89%, 100%, and 95%. The lower manufacturer accuracy for RF and LR, relative to their chipset accuracy, was caused by a single misclassified NIC (discussed below). Per-class precision, recall, and F1-score for both tasks and all classifiers are reported in Appendix B. These results provide evidence that, for the chipsets evaluated, NRAR profiles capture chipset-intrinsic behavior that generalizes across hosts and instances. Broader generalization requires evaluating a larger set of cross-device pairs, which we identify as a key direction for future work.

Two NICs accounted for most of the errors: the BCM4360 and the RTL8821AU. The BCM4360 was occasionally misclassified as a BCM4356 or a BCM4364, which is consistent with their similar n_{min} values and broadly similar NRAR profiles. Note that such misclassifications were also present in the evaluation of Section VI-B. Nonetheless, recall on this NIC remained at or above 26/30 (87%) across all three classifiers. The RTL8821AU proved more challenging: RF and LR frequently misclassified it as a Broadcom chip, with RF recall as low as 14/30 (47%). SVM, by contrast, correctly classified all 30 samples of this NIC.

VIII. Different Channel Conditions

Section VI and Section VII provide evidence that NRAR profiles can identify Wi-Fi NICs and that the discriminative signal is chipset-intrinsic rather than device-specific. Both evaluations, however, were conducted under favorable channel conditions (1m LOS) and within a short temporal window of the training data collection (at most one week). In this section, we evaluate two further assumptions on which NICPrint relies: NRAR profiles remain stable over time and degrade gracefully under channel impairment.

For this evaluation, we collected 30 NRAR profiles for each device in Table VI at the positions shown in Figure 9, for a total of 600 new NRAR profiles (30 NRAR profiles $\times$ 4 devices $\times$ 5 positions). As in Section VII, these new profiles are only used for testing classifiers trained with the 2,500 NRAR profiles from Section VI. The distances reported in Figure 9 are measured in a straight line from NICPrint's frame injector to the target device. The same set of devices was tested at every position, with measurements at each position collected on consecutive days over the course of one week. We collected all NRAR profiles in this section at least *one month* after the training set of Section VI. A description of each position follows.

Position 1. The target was located 3m from NICPrint's frame injector, behind a sheetrock wall. This represents an adversary attempting to identify NICs in a room they cannot enter.

Position 2. The target was 5m from NICPrint's frame injector and enclosed in a plastic box wrapped with six layers of aluminum foil, providing partial RF shielding (NLOS). This corresponds to identifying a NIC concealed inside a partially shielded container, approximating a Faraday bag.

Position 3. The target was 5 m from NICPrint's frame injector in a line-of-sight configuration. This models identifying a NIC at a moderate distance, without obstruction, in a room accessible to the adversary.

Position 4. The target was 8m from NICPrint's frame injector and inside a cardboard box (NLOS). This captures identifying a NIC inside a non-shielded enclosure at extended range.

Position 5. The target sat 1m from NICPrint's frame injector in line of sight, replicating the channel conditions of the training set. We measured this position last in the one-week collection window, making it the most temporally distant (one month and a week) from training data collection. Position 5 serves as the temporal-stability control: any variation relative to the profiles from Section VI reflects drift over the elapsed time rather than effects from channel impairment.

TABLE VI
DEVICES USED FOR EVALUATION UNDER VARIED CHANNEL CONDITIONS. NRAR PROFILES FOR EACH OF THESE DEVICES ARE CAPTURED AT EACH OF THE POSITIONS SHOWN IN FIGURE 9.

Device	Wi-Fi Chip	Manufacturer	Standard
Motorola G Power	MT6765	MediaTek	Wi-Fi 5
BrosTrend AC650	RTL8821AU	Realtek	Wi-Fi 5
Asus ROG	AC9560	Intel	Wi-Fi 5
MacBook Pro 2012	BCM4331	Broadcom	Wi-Fi 5

The devices selected for this evaluation are limited to those with portable form factors (smartphones, laptops, and USB dongles) that could be physically placed at each test position. Our testbed Qualcomm devices (Netgear Orbi units and M.2 modules with PCIe connection) require wired power, preventing placement behind walls or inside enclosures; we therefore did not include a Qualcomm NIC in this evaluation. The selection, however, included the `RTL8821AU`, which was the most challenging NIC in our cross-device evaluation (see Section VII), and the `BCM4331`, which clusters tightly with other Broadcom chips (see Appendix A). This subset therefore exercises both straightforward and challenging chipset-discrimination cases.

Figure 10a and Figure 10c report classifier accuracy at each of the five positions described above, for both the manufacturer and chipset classification tasks, respectively. Recall that in this evaluation we used the classifiers from Section VI, which are trained exclusively with NRAR profiles from the 25 devices in our testbed placed at 1m from NICPrint and in LOS configuration. Per-class precision, recall, and F1-score for all classifiers and positions are reported in Appendix C. The Position 5 results isolate the effect of the temporal gap from channel impairment. RF achieved 100% accuracy on both manufacturer and chipset classification while SVM achieved 98% on each. LR achieved 75% chipset identification accuracy and 73% manufacturer identification accuracy, which is a substantial drop compared to the same-session evaluation of Section VI. This drop is attributed to all samples of a single NIC (the `RTL8821AU`) being misclassified, while accuracy remains high for all others. This effect was also observed, to a lesser extent, in the cross-device evaluation of Section VII. Overall, the Position 5 results provide evidence that NRAR profiles do not measurably drift over the tested temporal gap (one month and one week) when channel conditions are held constant. For RF and SVM, classifier performance was essentially unchanged from the same-session evaluation of Section VI. Any degradation observed at the remaining positions can therefore be attributed to channel impairment rather than fingerprint instability over time.

Figure 10a reveals consistent patterns across Positions 1 to 4. First, SVM showed superior overall performance across positions. SVM was the only classifier whose manufacturer accuracy remained above 92% at every position. RF manufacturer accuracy ranged from 55% to 98%, and LR manufacturer accuracy ranged from 52% to 87%. The largest classifier-dependent gap appeared at Position 2, where SVM achieved 94% manufacturer accuracy while RF and LR fell to 55% and 52% respectively. This indicates that RF and LR are substantially more sensitive than SVM to the severe RF shielding. Note that Position 2 represents the worst-case scenario for NICPrint, since the aluminum-wrapped enclosure substantially degrades wireless communication. Furthermore, as shown in Figure 10c, chipset identification severely degraded at Position 2 for all classifiers. Overall, manufacturer accuracy is more robust to channel impairment than chipset accuracy. At Position 4, for example, RF manufacturer accuracy reached 98% while RF chipset accuracy dropped to 75%; LR showed the same trend at Position 4 (87% manufacturer accuracy vs. 52% chipset accuracy). This behavior is consistent with the vendor-clustering observation in Section V-C and motivates the two-stage classifier design we evaluate next.

A two-stage classifier architecture decomposes chipset identification into a manufacturer-identification stage followed by a manufacturer-specialized chipset identification stage. Chipset identification is the ultimate goal of this identification mechanism and the manufacturer stage exists to route the NRAR profile to the most appropriate specialized classifier. In the first stage, an ensemble of classifiers (RF, SVM, LR in our implementation) produces per-manufacturer confidence scores for a given NRAR profile. These scores are aggregated through soft voting where the inferred manufacturer $\hat{m}$ is the one that maximizes the sum of confidence scores across classifiers:

$$\hat{m} = \arg\max_{m} \sum_{i \in \{\text{RF, SVM, LR}\}} s_i(m \mid x),$$

where $s_i(m \mid x)$ is the confidence score classifier i assigns to manufacturer m for NRAR profile x. In the second stage, the identified manufacturer routes the NRAR profile to a manufacturer-specialized chipset classifier trained only on the chipsets of that manufacturer. We trained one specialized classifier per manufacturer for each of the three model families (RF, SVM, LR), using the same hyperparameter-tuning procedure described in Section V-D and the profiles from Section VI. Note that the final output of this stage is the chipset inference produced by the specialized classifier from the manufacturer identified in the first stage.

Figure 11 reports chipset accuracy of the two-stage classifier across the five positions. The two-stage classifier showed the largest improvement under moderate channel impairment (Positions 3 and 4). At Position 4, single-stage chipset accuracy ranged from 52% (LR) to 75% (RF); the two-stage architecture increased these accuracies to 74%, 84%, and 90% for LR, RF, and SVM, respectively. Similarly, at Position 3, RF chipset accuracy improved from 73% to 94%, while the spread between the best and worst classifier narrowed substantially. At Positions 1 and 5, the two-stage architecture performed comparably to the already accurate single-stage architecture. The results at Position 2 reveal a limitation of our two-stage approach. The soft-vote manufacturer accuracy at Position 2 was 79%, which sets a hard upper bound on chipset accuracy since any NRAR profile routed to the wrong specialized classifier cannot be correctly classified at the chipset level.

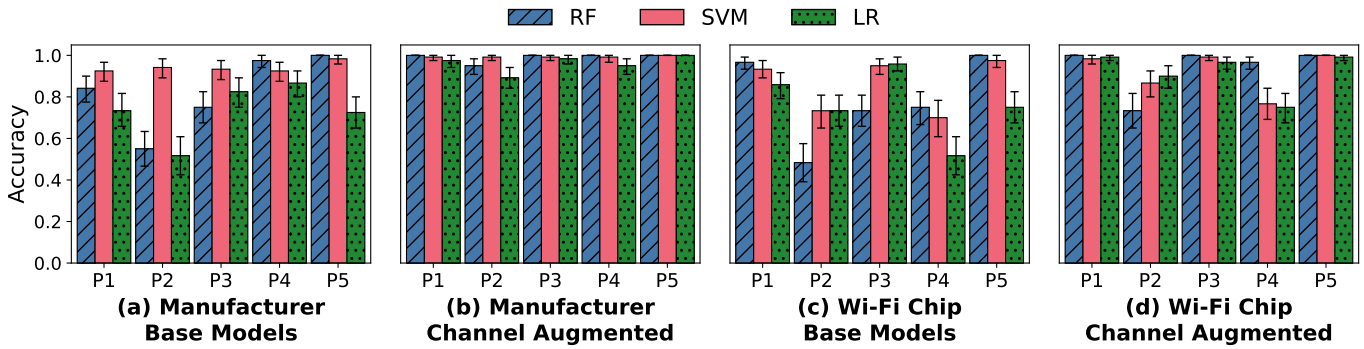


Fig. 10. Classifier accuracy for manufacturer and Wi-Fi chip identification across the five positions (P1-P5) of Figure 9, evaluated on NRAR profiles from the devices in Table VI: (a) and (c) show the accuracies for classifiers trained exclusively on the fixed-position dataset of Section VI (1m LOS) while (b) and (d) show the accuracies for classifiers trained with channel augmented data (see Section IX).

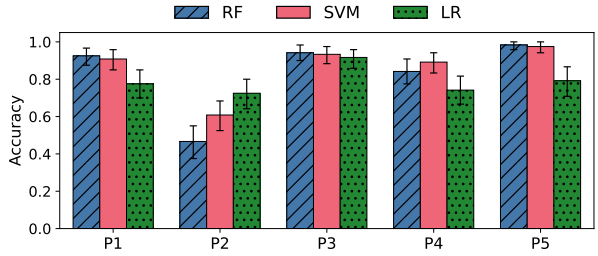


Fig. 11. Chipset classification accuracy of the two-stage classifier across the five positions of Figure 9, evaluated on NRAR profiles from the devices in Table VI. Manufacturer identification is performed via soft voting; chipset identification is then made by the manufacturer-specialized classifier. All classifiers are trained exclusively on the profiles from Section VI (1m LOS).

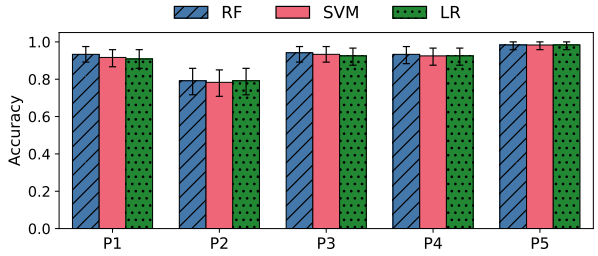


Fig. 12. Top-2 chipset accuracy of the two-stage classifier across the five positions of Figure 9, evaluated on NRAR profiles from the devices in Table VI. An inference is considered correct if the ground-truth chipset appears among the two highest-ranked outputs. All classifiers are trained exclusively on the profiles from Section VI (1m LOS).

Beyond single-inference chipset accuracy, we also examined top-2 chipset accuracy, defined as the fraction of NRAR profiles for which the correct chipset appears among the *two* highest-ranked inferences. This metric complements single-inference accuracy because reconnaissance can benefit from a tractable shortlist when a single inference cannot be accurately obtained; thus, making top-2 accuracy directly relevant to the adversary's targeting decisions. Figure 12 reports top-2 chipset accuracy for all positions. At Positions 1, 3, 4, and 5, top-2 chipset accuracy was at or above 90% for all classifiers. The improvement over single-inference accuracy was most pronounced at Position 4, where RF and LR rose from 84% and 74% to 93% each. At Position 2, top-2 accuracy was 79% for all classifiers, identical to the soft-vote manufacturer accuracy; this reaffirms the aforementioned limitation of the two-stage architecture. These results, together with the temporal-stability and channel robustness findings discussed earlier, characterize NICPrint's overall robustness. Across the conditions tested on a portable device subset, manufacturer and chipset identification of NICs remained stable over a month-long temporal gap. Moreover, the strongest classifier sustained manufacturer identification behind walls and at extended distances. Finally, the two-stage architecture sustains high accuracy chipset identification except in severely shielded environments.

IX. Channel Augmentation

Inspired by prior approaches to improve channel robustness in RF fingerprinting work, we evaluated whether channel diversity in the training set (i.e., channel augmentation) improves NICPrint robustness. We combined the 1m LOS dataset of Section VI with the multi-position dataset of Section VIII and retrain each classifier. Specifically, we use a leave-one-position-out procedure: for each position in Figure 9, every NRAR profile from that position is held out for testing and the classifier is trained on the remaining profiles of the combined dataset, with hyperparameters tuned by inner cross-validation on the training partition alone. Because the held-out position is never observed during training, classification results are for a previously unseen channel.

Figure 10b and Figure 10d show the manufacturer and chipset classification accuracy, respectively, when each position is held out for testing. Channel augmentation improves accuracy at every position and for every classifier compared to the results of the unaugmented models in Figure 10a and Figure 10c. The gains are largest at the positions where the unaugmented classifiers performed worst. On the manufacturer task at Position 2, RF rises from 55% to 95% and LR from 52% to 89%, while SVM (already the best performing classifier) improves from 94% to 99%. The chipset classification task

shows the same pattern at the two hardest positions: at Position 2, RF improves from 48% to 73% and LR from 73% to 90%; similarly, at Position 4, RF improves from 75% to 97% and LR from 52% to 75%. At Positions 1, 3, and 5, where the unaugmented classifiers were already accurate, augmentation raises chipset accuracy to 100% for RF and to at least 97% for SVM and LR. Averaged across the five positions and three classifiers, channel augmentation raises chipset accuracy from 80% to 93% and manufacturer accuracy from 83% to 98%. Moreover, the spread between the best and worst position narrows from 45 to 5 percentage points for RF and from 35 to 11 percentage points for LR on the manufacturer task. Note that RF at Position 2 along with SVM and LR at Position 4 are the only cases where chipset accuracy drops below 90%. In each, the errors are dominated by the `RTL8821AU`, which has proven difficult to classify accurately across all evaluations. Manufacturer accuracy at those same positions remains at 95% or better, so the augmented classifiers still attribute these profiles to the correct manufacturer even when they cannot accurately identify the specific chipset. Note that channel augmentation yields a larger performance improvement than the two-stage classifier described in Section VIII. Channel augmentation, however, *requires* a channel-diverse training set; thus, the preferred approach may depend on the adversary's constraints when creating the reference set.

X. Defenses

NICPrint exploits implementation diversity in PHY basebands of Wi-Fi NICs combined with the polite-ACK behavior that makes this diversity remotely observable. We outline three defense directions, targeting the PHY layer, the MAC-layer ACK behavior, and detection of the identification mechanism.

PHY-layer implementation standardization. NRAR profiles differ across vendors because of differences in PHY baseband designs; the precise sources of variation, however, are proprietary and not directly observable. A defense that constrains the space of acceptable PHY-layer implementations – for example, by mandating that all standard-compliant receivers exhibit STF tolerance within a narrow range – would shrink the NRAR feature space without proscribing exact baseband designs.

ACK rate-limiting. Wi-Fi NICs could rate-limit ACK responses to senders that are not part of the current association context, or whose recent transmission patterns exhibit characteristics consistent with active probing (e.g., sustained transmission with varying preamble characteristics). Note that this defense also addresses other polite-ACK related exploits.

Defender-side detection. A wireless monitoring system positioned within range of a defended environment could detect NICPrint's transmissions. Detecting individual malformed STFs is complicated because naturally occurring malformed frames can result from severe multipath or other channel impairments. Nonetheless, the sustained transmission pattern that NICPrint relies on, where n is swept in a systematic way, cannot be attributed to channel effects alone. We expect this to be the most rapidly deployable countermeasure since it requires no changes to the NICs of client devices.

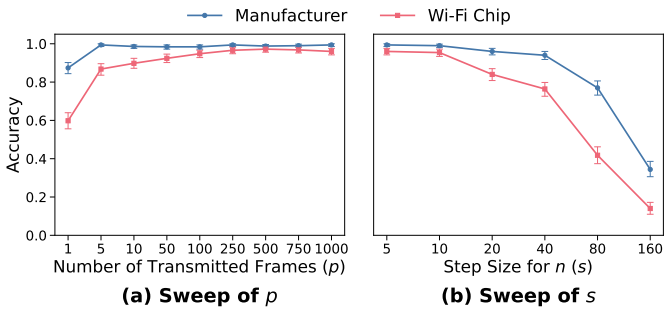


Fig. 13. Chipset and manufacturer classification accuracy for (a) different transmitted frames p per number of invalid samples in the STF and (b) different step sizes s in the NRAR profile.

XI. Discussion and Limitations

In this section, we discuss the operational scope of NICPrint, alternative system parameter choices, and the limitations of the proposed technique.

Reference-set acquisition. NICPrint requires the adversary to build a labeled reference set of NRAR profiles (profiling phase in Section V), which assumes that the adversary knows the chipset of each training device. For devices running general-purpose operating systems (OS), such as smartphones and laptops, chipset identification is trivial through standard system programs (e.g., `lspci`, `iw`). For embedded devices, however, chipset identification may require physical teardown, FCC ID lookup, or firmware analysis.

MAC randomization. Modern OSes randomize MAC addresses, limiting identification techniques that rely on repeated probing of a single MAC address. NICPrint operates within the constraints of existing randomization policies, which rotate MAC addresses upon network association while keeping them static during an active connection. iOS, Android, Windows, and macOS all preserve MAC stability across the lifetime of a single connection, which typically spans minutes to hours. At our default parameters ($p = 1000$ and $s = 5$), the active transmission time required to construct a complete NRAR profile is approximately 33 seconds, with NRAR computation performed offline after collection. The next paragraph shows that this transmission time can be reduced substantially.

Profiling time and frame budget. NICPrint is not stealthy in terms of transmission volume. Nonetheless, as noted in Section V, malformed frames are discarded at the target's PHY and never surface to the host OS; thus, detection requires a monitoring system covering the Wi-Fi channel rather than relying on host-side artifacts. Moreover, our default settings require approximately 33 seconds of transmission time and 33,000 transmitted frames per profile. Lower values of p (number of transmitted frames for each n) and larger values of s (step size for the number of zeroed STF samples) reduce profiling time and improve stealthiness at the cost of accuracy. Figure 13a reports SVM manufacturer and chipset classification accuracy as a function of p, with s fixed at the default of 5, evaluated with a single stratified 80/20 split on the dataset of

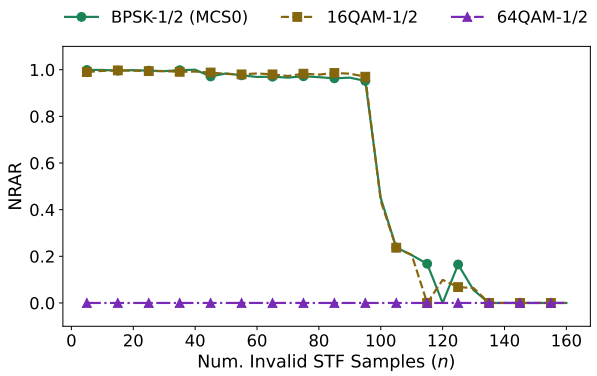


Fig. 14. NRAR profiles for the `BCM4345C0` when frames are transmitted with BPSK-1/2, 16QAM-1/2, and 64QAM-1/2. The Wi-Fi NIC is located 8 meters from frame injector and inside a cardboard box (NLOS).

Section VI. For each value of p, we resampled the underlying ACK measurements, recomputed NRAR profiles, and re-tuned hyperparameters. We report SVM only because it exhibited the most robust performance across all evaluations, but similar trends occur with the other tested classifiers (i.e., RF and LR). Notice that accuracy remains above 95% for both chipset and manufacturer identification at $p \geq 250$. Figure 13b shows the analogous sweep over s, where accuracy degrades sharply for $s > 10$. Combining both reductions, $p = 250$ with $s = 10$, corresponds to approximately 4,000 transmitted frames per profile (or roughly 4 seconds of transmission time), an 8x reduction relative to the default configuration, while sustaining accuracy in the high 90s. Note that an adversary could also spread these transmissions over longer periods of time rather than transmit continuously, as a mean to increase stealthiness.

MCS selection. Section V-D specified BPSK-1/2 as the default modulation and coding scheme (MCS) used for all transmitted frames. The choice of MCS does not appear to affect NRAR profiles under benign channel conditions. We constructed NRAR profiles from transmissions with higher MCSs, and they exhibited similar behavior to their BPSK-1/2 counterparts. Under impaired channel conditions, however, the choice of MCS becomes important (see Figure 14). With higher MCS values, frames are more likely to fail decoding for reasons unrelated to the STF malformation, leading to ACK loss that degrades the NRAR signal we are trying to measure. BPSK-1/2 maximizes the decoding likelihood and isolates ACK loss to the STF-malformation effect. We therefore use BPSK-1/2 as the default, but highlight that profiles can in principle be constructed at any MCS supported by both injector and target.

NICs with shared PHY designs. NRAR profiles capture implementation-specific behavior of the receiver's PHY baseband design. When two chipsets share this design to a large extent, as appears to be the case for the `QCA6174` and `QCA9377` (see Appendix A), NICPrint cannot distinguish them. NICPrint's effective resolution is therefore "PHY-pipeline family" rather than strictly "chipset model"; the distinction depends on vendor design practices.

Open-set classification. NICPrint uses closed-set classifiers. The classifiers always map an input NRAR profile to one of the classes in the training set; hence, attempting to identify a target whose chipset is not in the reference set will always result in an incorrect inference. In real deployments, this can be partially mitigated through confidence thresholding. Moreover, adversaries with reference sets covering commonly deployed chipsets will rarely encounter this issue, whereas those operating in environments with niche hardware may encounter it more frequently.

Evaluation scope. The cross-device evaluation of Section VII encompasses five devices, one from each manufacturer in our testbed. The sample is sufficient to establish that NRAR profiles exhibit consistent structure, but it does not provide strong evidence for broad generalization. Expanding this evaluation requires acquiring multiple host platforms that use the same chipset. Since many chipsets are only available in consumer products, obtaining distinct hosts with the same chipset often requires purchasing complete consumer devices. Moreover, the channel-condition evaluation of Section VIII excludes Qualcomm devices in our testbed because they require AC power or PCIe connection, preventing us from physically positioning them at the test locations. Qualcomm chipsets exhibited the lowest within-vendor separability in the evaluation of Section VI; therefore, NICPrint's channel robustness for the most challenging vendor remains uncharacterized.

Probe scope. NICPrint corrupts only the STF. The LTF is a natural complementary target since it drives a distinct part of the PHY baseband pipeline (i.e., channel and fine CFO estimation). We restricted our probing to the STF for two reasons. First, corrupting the LTF degrades channel estimation, which severely impacts frame decodability; thus, the resulting ACK losses cannot be attributed exclusively to implementation-specific behavior, especially under difficult channel conditions. Second, LTF processing exhibited less implementation variation in the open-source stacks we examined. Nonetheless, we regard LTF exploration as promising future work.

XII. Summary

NICPrint shows that the way Wi-Fi NICs respond to malformed preambles can identify their manufacturer and chipset model. By transmitting frames with controlled preamble corruption and observing polite-ACK responses, an adversary can construct a fingerprint of the target NIC without I/Q access, CSI, or target cooperation beyond ACK transmission. We evaluated NICPrint on 25 devices spanning five major Wi-Fi vendors and four Wi-Fi generations. On previously unseen devices with chipsets represented in training, NICPrint averages 95% manufacturer and 96% chipset accuracy across the tested classifiers. Under non-line-of-sight, through-wall, and extended-distance conditions measured more than one month later, the base classifiers average 83% manufacturer and 80% chipset accuracy; channel augmentation increases these to 98% and 93%, respectively. Even without augmentation, a proposed two-stage classifier places the correct chipset in the top two predictions with at least 90% accuracy outside severe shielding.

Acknowledgements

We thank the anonymous reviewers for their valuable suggestions. We are grateful to Dr. Sergey Bratus for motivating us to pursue this investigation. We used AI tools to improve the grammar and clarity of human-authored text and to refine the code generating the plots and tables to improve their presentation. We (the human authors) designed the system, developed the evaluations, and analyzed the results. This research results from a research program at the Institute for Security, Technology, and Society at Dartmouth College, supported by the National Science Foundation under award number CNS-1955805. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors. Any mention of specific companies or products does not imply any endorsement by the authors, by their employers, or by the sponsors.

Ethics Considerations

We analyze the ethical implications of this work for three stakeholder groups: the research team, Wi-Fi users whose devices could be identified by NICPrint, and Wi-Fi chipset vendors and implementers.

Research team. This work involved only controlled laboratory experiments conducted by us (the authors). We owned and operated all Wi-Fi devices and networks used in this investigation. We did not attempt to fingerprint any device belonging to third parties. Our experiments did not involve human participants and we did not collect sensitive data. All wireless transmissions complied with FCC regulations, and we conducted all experiments in frequencies with limited occupancy to mitigate interference with legitimate Wi-Fi traffic.

Wi-Fi users. Identification of a device Wi-Fi chipset can enable targeted exploitation and could, in some cases, narrow the search space sufficiently to allow tracking. Several factors mitigate this risk. First, the discriminative behavior NICPrint exploits is observable in open-source Wi-Fi stacks (see Section V-D); a motivated adversary could have extended this observation to commercial chipsets independently. Second, chipset identification is meaningful only in combination with chipset-specific exploits, which are independently disclosed and patched. Third, we describe defenses in Section X.

Wi-Fi chipset vendors and implementers. Our findings show that vendor-specific implementation choices in the PHY baseband expose a chipset-discriminative signal that is observable over the air. Wi-Fi chipset manufacturers could use this investigation as a basis for developing PHY-layer designs that reduce the distinctiveness of this signal. Moreover, our results might help the 802.11 Working Group consider further standardization that limits implementation diversity. Although these efforts impose financial and development costs on implementers, we provide detailed technical descriptions of the discriminating mechanism (i.e., NRAR profiles) as well as all of our code and data to support further investigations.

References

[1] R. Mangar, T. J. Pierson, and D. Kotz, "A Framework for Evaluating the Security and Privacy of Smart-Home Devices, and its Application to Common Platforms," *IEEE Pervasive Computing*, vol. 23, no. 3, Jul. 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10627940/

[2] Markets and Markets, "Wi-Fi Chipset Market Size, Share & Trends, 2030," Aug. 2025. [Online]. Available: https://www.marketsandmarkets.com/Market-Reports/wi-fi-chipset-market-42186393.html

[3] Mordor Intelligence, "Wi-Fi Chipset Market Size & Share Analysis — Growth Trends and Forecast (2026–2031)," Jan. 2026. [Online]. Available: https://www.mordorintelligence.com/industry-reports/global-wi-fi-chipset-market

[4] M. Čermák, S. Svorenčík, and R. Lipovský, "Kr00k – CVE-2019-15126: A Serious Vulnerability Deep Inside Wi-Fi Encryption," Feb. 2020. [Online]. Available: https://web-assets.esetstatic.com/wls/2020/02/ESET_Kr00k.pdf

[5] N. Artenstein, "Broadpwn: Remotely Compromising Android and iOS via a Bug in Broadcom's WiFi Chipsets," Jul. 2017. [Online]. Available: https://blackhat.com/docs/us-17/thursday/us-17-Artenstein-Broadpwn-Remotely-Compromising-Android-And-iOS-Via-A-Bug-In-Broadcoms-Wifi-Chipsets-wp.pdf

[6] M. Vanhoef, "Fragment and Forge: Breaking Wi-Fi Through Frame Aggregation and Fragmentation," in *Proceedings of the USENIX Security Symposium*. USENIX Association, Aug. 2021. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/vanhoef

[7] H. Cao, L. Huang, S. Hu, S. Shi, and Y. Liu, "Owfuzz: Discovering Wi-Fi Flaws in Modern Devices through Over-The-Air Fuzzing," in *Proceedings of the ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*. ACM, May 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3558482.3590174

[8] J. Franklin and D. McCoy, "Passive Data Link Layer 802.11 Wireless Device Driver Fingerprinting," in *Proceedings of the USENIX Security Symposium*. USENIX Association, Jul. 2006. [Online]. Available: https://www.usenix.org/conference/15th-usenix-security-symposium/passive-data-link-layer-80211-wireless-device-driver

[9] P. Robyns, B. Bonné, P. Quax, and W. Lamotte, "Noncooperative 802.11 MAC Layer Fingerprinting and Tracking of Mobile Devices," *Security and Communication Networks*, vol. 2017, no. 1, 2017. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2017/6235484

[10] C. Neumann, O. Heen, and S. Onno, "An Empirical Study of Passive 802.11 Device Fingerprinting," in *Proceedings of the International Conference on Distributed Computing Systems Workshops*. IEEE Computer Society, 2012. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICDCSW.2012.8

[11] T. Kohno, A. Broido, and K. C. Claffy, "Remote Physical Device Fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, 2005. [Online]. Available: https://doi.org/10.1109/TDSC.2005.26

[12] S. Jana and S. K. Kasera, "On Fast and Accurate Detection of Unauthorized Wireless Access Points Using Clock Skews," *IEEE Transactions on Mobile Computing*, vol. 9, no. 3, 2010. [Online]. Available: https://doi.org/10.1109/TMC.2009.145

[13] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the Reliability of Wireless Fingerprinting using Clock Skews," in *Proceedings of the ACM Conference on Wireless Network Security (WiSec)*. ACM, March 2010. [Online]. Available: https://doi.org/10.1145/1741866.1741894

[14] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless Device Identification with Radiometric Signatures," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*. ACM, 2008. [Online]. Available: https://doi.org/10.1145/1409944.1409959

[15] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized Radio Classification through Convolutional Neural Networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2019. [Online]. Available: https://doi.org/10.1109/INFOCOM.2019.8737463

[16] F. Restuccia, S. D'Oro, A. Al-Shawabka, M. Belgiovine, L. Angioloni, S. Ioannidis, K. Chowdhury, and T. Melodia, "DeepRadioID: Real-Time Channel-Resilient Optimization of Deep Learning-Based Radio Fingerprinting Algorithms," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and*

Computing (MobiHoc). ACM, 2019. [Online]. Available: https://doi.org/10.1145/3323679.3326503

[17] R. Kong and H. Chen, "CSI-RFF: Leveraging Micro-Signals on CSI for RF Fingerprinting of Commodity WiFi," *IEEE Transactions on Information Forensics and Security*, vol. 19, 2024. [Online]. Available: https://doi.org/10.1109/TIFS.2024.3396375

[18] ——, "DeepCRF: Deep Learning-Enhanced CSI-Based RF Fingerprinting for Channel-Resilient WiFi Device Identification," *IEEE Transactions on Information Forensics and Security*, vol. 20, Dec. 2024. [Online]. Available: https://doi.org/10.1109/TIFS.2024.3515796

[19] A. Abedi and O. Abari, "WiFi Says "Hi!" Back to Strangers!" in *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*. ACM, Nov. 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3422604.3425951

[20] *IEEE Standard for Information Technology — Telecommunications and Information Exchange Between Systems — Local and Metropolitan Area Networks — Specific Requirements — Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std. IEEE 802.11-2012, Mar. 2012. [Online]. Available: https://standards.ieee.org/ieee/802.11/4523/

[21] T. Schmidl and D. Cox, "Robust Frequency and Timing Synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, no. 12, Dec. 1997. [Online]. Available: http://ieeexplore.ieee.org/document/650240/

[22] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles, "Active Behavioral Fingerprinting of Wireless Devices," in *Proceedings of the ACM Conference on Wireless Network Security (WiSec)*. ACM, Mar. 2008. [Online]. Available: https://dl.acm.org/doi/10.1145/1352533.1352543

[23] K. Gopinath, P. Bhagwat, and K. Gopinath, "An Empirical Analysis of Heterogeneity in IEEE 802.11 MAC Protocol Implementations and its Implications," in *Proceedings of the International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*. ACM, Sep. 2006. [Online]. Available: https://doi.org/10.1145/1160987.1161002

[24] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens, "Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms," in *Proceedings of the ACM Asia Conference on Computer and Communications Security (Asia CCS)*. ACM, May 2016. [Online]. Available: https://doi.org/10.1145/2897845.2897883

[25] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "GTID: A Technique for Physical Device and Device Type Fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, 2015. [Online]. Available: https://doi.org/10.1109/TDSC.2014.2369033

[26] D.-J. Huang, K.-T. Yang, C.-C. Ni, W.-C. Teng, T.-R. Hsiang, and Y.-J. Lee, "Clock Skew Based Client Device Identification in Cloud Environments," in *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, Apr. 2012. [Online]. Available: https://doi.org/10.1109/AINA.2012.51

[27] A. Elmaghbub and B. Hamdaoui, "No Blind Spots: On the Resiliency of Device Fingerprints to Hardware Warm-Up Through Sequential Transfer Learning," in *Proceedings of the ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*. ACM, May 2024. [Online]. Available: https://doi.org/10.1145/3643833.3656138

[28] S. Alhazbi, S. Sciancalepore, and G. Oligeri, "The Day-After-Tomorrow: On the Performance of Radio Fingerprinting over Time," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*. ACM, Dec. 2023. [Online]. Available: https://doi.org/10.1145/3627106.3627192

[29] C. Matte, M. Cunche, F. Rousseau, and M. Vanhoef, "Defeating MAC Address Randomization Through Timing Attacks," in *Proceedings of the ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec)*, Jul. 2016. [Online]. Available: https://doi.org/10.1145/2939918.2939930

[30] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown, "A Study of MAC Address Randomization in Mobile Devices and When it Fails," in *Proceedings on Privacy Enhancing Technologies (PoPETs)*, vol. 2017, no. 4, 2017. [Online]. Available: https://doi.org/10.1515/popets-2017-0054

[31] L. Polcak and B. Frankova, "Clock-Skew-Based Computer Identification: Traps and Pitfalls," *Journal of Universal Computer Science*, vol. 21, no. 9, 2015. [Online]. Available: https://doi.org/10.3217/jucs-021-09-1210

[32] C. Wang, W. He, T. J. Pierson, and D. Kotz, "Moat: Adaptive Inside/Outside Detection System for Smart Homes," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 8, no. 4, Nov. 2024. [Online]. Available: https://dl.acm.org/doi/10.1145/3699751

[33] A. Jagannath, J. Jagannath, and P. S. P. V. Kumar, "A Comprehensive Survey on Radio Frequency (RF) Fingerprinting: Traditional Approaches, Deep Learning, and Open Challenges," *Computer Networks*, vol. 219, Dec. 2022. [Online]. Available: https://doi.org/10.1016/j.comnet.2022.109455

[34] N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More Is Better: Data Augmentation for Channel-Resilient RF Fingerprinting," *IEEE Communications Magazine*, vol. 58, no. 10, Nov. 2020. [Online]. Available: https://doi.org/10.1109/MCOM.001.2000180

[35] I. R. Jenkins, R. Shapiro, S. Bratus, T. Goodspeed, R. Speers, and D. Dowd, "Speaking the Local Dialect: Exploiting Differences Between IEEE 802.15.4 Receivers with Commodity Radios for Fingerprinting, Targeted Attacks, and WIDS Evasion," in *Proceedings of the ACM Conference on Security and Privacy in Wireless & Mobile Networks (WiSec)*. ACM, Jul. 2014. [Online]. Available: https://dl.acm.org/doi/10.1145/2627393.2627408

[36] B. Bloessl, C. Sommer, F. Dressler, and D. Eckhoff, "The Scrambler Attack: A Robust Physical Layer Attack on Location Privacy in Vehicular Networks," in *Proceedings of the IEEE International Conference on Computing, Networking and Communications (ICNC)*, Mar. 2015. [Online]. Available: https://doi.org/10.1109/ICCNC.2015.7069376

[37] Z. Cai, M. Krunz, and H. Rahbari, "Preamble Injection and Spoofing Attacks in Wi-Fi Networks," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Feb. 2021. [Online]. Available: https://doi.org/10.1109/GLOBECOM46510.2021.9685461

[38] B. Bloessl, "IEEE 802.11 Transceiver for GNU Radio," Sep. 2025. [Online]. Available: https://github.com/bastibl/gr-ieee802-11

[39] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman, "OpenWifi: a Free and Open-Source IEEE 802.11 SDR Implementation on SoC," in *Proceedings of the IEEE Vehicular Technology Conference*. IEEE, May 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9128614/

[40] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 Packet Delivery from Wireless Channel Measurements," in *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM) conference*. ACM, Aug. 2010. [Online]. Available: https://dl.acm.org/doi/10.1145/1851182.1851203

[41] C. Arguello, B. Perez, T. J. Pierson, and D. Kotz, "NICPrint: Wi-Fi NIC Identification via Malformed Preambles (Artifact)," Oct. 2026. [Online]. Available: https://doi.org/10.5281/zenodo.23508520

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, 2011. [Online]. Available: http://jmlr.org/papers/v12/pedregosa11a.html

[43] scikit-learn Developers, *GridSearchCV*, scikit-learn, 2026, scikit-learn 1.8.0 documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[44] U. Michelucci and F. Venturini, "Estimating Neural Network's Performance with Bootstrap: A Tutorial," *Machine Learning and Knowledge Extraction*, vol. 3, no. 2, Mar. 2021. [Online]. Available: https://www.mdpi.com/2504-4990/3/2/18

[45] S. Raschka, "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning," 2018. [Online]. Available: https://arxiv.org/abs/1811.12808

APPENDIX A
EXTENDED DISCUSSION OF SAME SESSION EVALUATION

`QCA6174` and `QCA9377` achieved LR F1-scores of 0.70 and 0.69, respectively, and were frequently confused with one another. Their NRAR profiles have similar n_{min} values and sharp degradation right after (see Figure 16). Furthermore, the cosine similarity between their NRAR profiles is 0.99, demonstrating that their patterns are indistinguishable. These observations are consistent with partial reuse of PHY basebands and illustrate a limitation of NICPrint: chipsets with highly similar baseband behavior may not be distinctly identifiable. A second intra-vendor cluster appears among some Broadcom

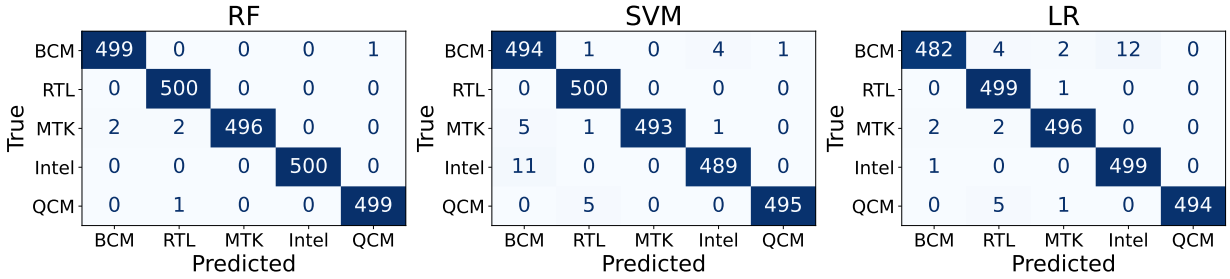


Fig. 15. Confusion matrices from Section VI-A evaluation. Class names: Broadcom (BCM), Realtek (RTL), MediaTek (MTK), and Qualcomm (QCM).

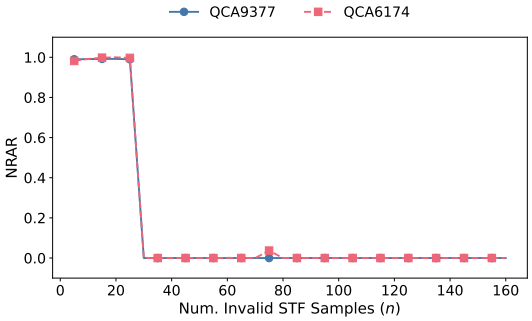


Fig. 16. NRAR profiles of QCA6174 and QCA9377 chipsets.

chipsets. BCM4356 and BCM4360 achieved SVM F1-scores of 0.84 and 0.88, and LR F1-scores of 0.82 and 0.85. Furthermore, the BCM4360 drops recall to 0.84 under SVM and 0.79 under LR, and the BCM4331 dropped recall to 0.84 under SVM. This pattern is consistent with the n_{min} clustering in Figure 4. Note that the BCM4331, BCM4356, BCM4360, and BCM4364 all exhibit n_{min} values within a narrow band ($80 \leq n_{min} \leq 95$), and their NRAR profiles share similar step-like degradation past n_{min} (see Figure 5b).

Appendix B
Per-class Metrics for Cross-device Evaluation

Table VII and Table VIII report per-class precision, recall, and F1-score for chipset and manufacturer identification, respectively, in the cross-device evaluation (Section VII). The RTL8821AU accounted for most errors: RF achieved a chipset recall of 0.67 and a Realtek manufacturer recall of 0.47. This chipset consistently degraded performance across all evaluations. Moreover, the BCM4360 showed the same intra-vendor ambiguity described in Appendix A. The errors, however, remained within the Broadcom class, so manufacturer recall was 1.00 across all classifiers. This represents the most favorable error case, where vendor inference succeeds while the residual chipset ambiguity is confined to the chipset family.

Appendix C
Per-class metrics for Unseen Channel Evaluation

Tables IX and X report per-class precision, recall, and F1-score for all classifiers and positions in Section VIII. For manufacturer classification, most errors involved the Realtek class. At Position 2, RF and LR achieved recall of 0.13 and 0.30 respectively on this class, while at Position 3 RF recall dropped to 0.00. Intel classification also degraded at Position 2 for RF (recall 0.07) and LR (recall 0.00). We believe this is somewhat expected given the severe degradation of wireless communication in the aluminum-shielded enclosure. For chipset classification, the dominant failure was the RTL8821AU, consistent with the cross-device findings of Section VII. RF achieved zero recall on this chipset at Positions 2, 3, and 4. SVM and LR also struggled, with RTL8821AU recall dropping to 0.07 (SVM, Position 4) and 0.03 (LR, Position 4). The MT6765 showed poor performance only for LR; recall dropped to 0.13 at Position 4 and 0.17 at Position 5, indicating that LR has difficulty maintaining this chipset's profile signature even under benign conditions.

TABLE VII
Per-class precision, recall, and F1-score for Wi-Fi chip classification of NICs in Table V (previously unseen devices).

	RF			SVM			LR		
Chipset	P	R	F1	P	R	F1	P	R	F1
BCM4360	1.00	1.00	1.00	1.00	0.93	0.97	1.00	0.87	0.93
RTL8821AU	1.00	0.67	0.80	1.00	1.00	1.00	1.00	1.00	1.00
MT7961	1.00	0.97	0.98	1.00	1.00	1.00	0.97	1.00	0.98
AC8265	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
QCA9377	1.00	0.97	0.98	1.00	1.00	1.00	1.00	1.00	1.00
Avg	1.00	0.92	0.95	1.00	0.99	0.99	0.99	0.97	0.98

Uncertainties: the median 95% CI half-width is 0.00 for all three classifiers, with maxima of 0.16/0.09/0.11 for RF/SVM/LR; the widest interval is RF recall on RTL8821AU.

TABLE VIII
Per-class precision, recall, and F1-score for manufacturer classification of NICs in Table V (previously unseen devices).

	RF			SVM			LR		
Vendor	P	R	F1	P	R	F1	P	R	F1
BCM	0.65	1.00	0.79	1.00	1.00	1.00	1.00	1.00	1.00
RTL	1.00	0.47	0.64	1.00	1.00	1.00	1.00	0.77	0.87
MTK	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Intel	1.00	1.00	1.00	1.00	1.00	1.00	0.94	1.00	0.97
QCM	1.00	1.00	1.00	1.00	1.00	1.00	0.86	1.00	0.92
Avg	0.93	0.89	0.89	1.00	1.00	1.00	0.96	0.95	0.95

Uncertainties: the median 95% CI half-width is 0.00 for all three classifiers, with maxima of 0.18/0.00/0.15 for RF/SVM/LR; the maxima are from RF/LR recall and RF F1 on RTL.

TABLE IX
PER-POSITION AND PER-CLASS PRECISION, RECALL, AND F1-SCORE FOR MANUFACTURER CLASSIFICATION.

Pos.	Class	RF			SVM			LR		
		P	R	F1	P	R	F1	P	R	F1
P1	BCM	0.61	1.00	0.76	1.00	0.90	0.95	0.92	0.80	0.86
	RTL	1.00	0.37	0.54	0.83	1.00	0.91	0.75	0.30	0.43
	MTK	1.00	1.00	1.00	1.00	0.80	0.89	0.96	0.90	0.93
	Intel	1.00	1.00	1.00	0.91	1.00	0.95	0.76	0.93	0.84
P2	BCM	0.36	1.00	0.53	0.96	0.90	0.93	0.43	0.77	0.55
	RTL	1.00	0.13	0.24	1.00	0.87	0.93	1.00	0.30	0.46
	MTK	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Intel	1.00	0.07	0.13	0.86	1.00	0.92	0.00	0.00	0.00
P3	BCM	0.50	1.00	0.67	1.00	0.83	0.91	0.96	0.83	0.89
	RTL	0.00	0.00	0.00	0.91	1.00	0.95	0.90	0.57	0.69
	MTK	1.00	1.00	1.00	1.00	0.90	0.95	1.00	0.93	0.97
	Intel	1.00	1.00	1.00	0.86	1.00	0.92	0.85	0.97	0.91
P4	BCM	0.94	0.97	0.95	0.96	0.87	0.91	1.00	0.80	0.89
	RTL	1.00	0.93	0.97	0.88	0.97	0.92	0.86	0.80	0.83
	MTK	1.00	1.00	1.00	1.00	0.87	0.93	0.81	0.87	0.84
	Intel	0.97	1.00	0.98	0.88	1.00	0.94	0.83	1.00	0.91
P5	BCM	1.00	1.00	1.00	1.00	0.97	0.98	0.97	0.97	0.97
	RTL	1.00	1.00	1.00	0.97	1.00	0.98	0.00	0.00	0.00
	MTK	1.00	1.00	1.00	1.00	0.97	0.98	1.00	0.97	0.98
	Intel	1.00	1.00	1.00	0.97	1.00	0.98	0.97	0.97	0.97
All	**Avg**	**0.87**	**0.82**	**0.79**	**0.95**	**0.94**	**0.94**	**0.80**	**0.73**	**0.75**

Uncertainties: median 95% CI half-widths are 0.00/0.05/0.07 for RF/SVM/LR, with maxima of 0.50/0.15/0.27. The 0.50 outlier is RF precision for Intel at Pos. 2.

TABLE X
PER-POSITION AND PER-CLASS PRECISION, RECALL, AND F1-SCORE FOR WI-FI CHIP CLASSIFICATION.

Pos.	Class	RF			SVM			LR		
		P	R	F1	P	R	F1	P	R	F1
P1	BCM4331	1.00	1.00	1.00	1.00	0.90	0.95	1.00	1.00	1.00
	RTL8821AU	1.00	0.87	0.93	1.00	1.00	1.00	1.00	1.00	1.00
	MT6765	1.00	1.00	1.00	1.00	0.83	0.91	1.00	0.50	0.67
	AC9560	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.93	0.97
P2	BCM4331	0.49	0.57	0.52	0.91	0.67	0.77	0.94	0.57	0.71
	RTL8821AU	0.00	0.00	0.00	1.00	0.37	0.54	1.00	0.37	0.54
	MT6765	1.00	0.37	0.54	1.00	0.90	0.95	0.97	1.00	0.98
	AC9560	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P3	BCM4331	0.48	0.93	0.64	1.00	0.97	0.98	1.00	1.00	1.00
	RTL8821AU	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00
	MT6765	1.00	1.00	1.00	1.00	0.83	0.91	1.00	0.83	0.91
	AC9560	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P4	BCM4331	0.50	1.00	0.67	0.91	0.97	0.94	0.87	0.90	0.89
	RTL8821AU	0.00	0.00	0.00	1.00	0.07	0.13	1.00	0.03	0.07
	MT6765	1.00	1.00	1.00	1.00	0.77	0.87	0.80	0.13	0.23
	AC9560	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P5	BCM4331	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.93	0.97
	RTL8821AU	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	MT6765	1.00	1.00	1.00	1.00	0.90	0.95	1.00	0.17	0.29
	AC9560	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.90	0.95
All	**Avg**	**0.77**	**0.79**	**0.76**	**0.99**	**0.86**	**0.89**	**0.98**	**0.76**	**0.81**

Uncertainties: median 95% CI half-width is 0.00 for all classifiers, with maxima of 0.20/0.50/0.50 for RF/SVM/LR. The max is from SVM/LR P on RTL8821AU at Pos. 4.

APPENDIX D
CHIPSET DISTINCTNESS

Our findings rely on the assumption that the tested chipsets have distinct baseband designs. Although our experiments suggest this is true in most cases, publicly available information does not allow us to verify it. The finest granularity we can establish is therefore *silicon distinctness*, which does not necessarily imply baseband distinctness and does not guarantee that probed stages are not reused. We treat chipsets from different manufacturers as silicon-distinct and focus the analysis below on intra-manufacturer distinctness. Within a manufacturer, chipsets from different Wi-Fi generations must use distinct silicon to support newly introduced PHY functionality; we refer to this as the *generation criterion* and apply it to every pair not discussed explicitly below.

Broadcom: Broadcom chips expose a chip ID, revision number, and other flags through the ChipCommon core, with distinct IDs corresponding to distinct silicon designs. We verified that each Broadcom chip in our testbed reports a unique ID. Furthermore, for chips with publicly available datasheets (`BCM43455` and `BCM4356`), we confirmed differences in SRAM and ROM sizes, which require distinct silicon designs.

Realtek: Unlike the other manufacturers, the identifiers laser-printed on Realtek chip packages match the marketing identifiers used in this paper, suggesting that different chipset identifiers correspond to distinct silicon. The `RTL8822` also supports Wave 2 functionality,[4] whereas its same-generation counterparts are Wave 1. In addition, each Wi-Fi 5 chipset has firmware images of different sizes, and chipsets with larger firmware images also expose larger TX/RX hardware FIFO buffers in the `hw_spec` structure of the Linux `rtw88` driver. Together, these differences suggest distinct on-die memory configurations and therefore distinct silicon designs.

MediaTek: The `MT6765` is an SoC and therefore uses distinct silicon from its same-generation counterparts in our testbed. We could not establish silicon distinctness for the `MT7921`/`MT7961` pair. All other pairs follow from the generation criterion.

Intel: The `AC9560` and `AX201` are Connectivity Integration (CNVi)[5] modules and therefore use distinct silicon from their same-generation counterparts. The `AC8265` is advertised as supporting Wave 2 functionality, whereas the `AC3160` is a Wave 1 device. Depending on which Wave 2 features are implemented, this provides evidence of silicon distinctness. Moreover, the `AC8265` firmware image is two times larger than that of the `AC3160`, suggesting different on-die memory requirements and, consequently, distinct silicon.

Qualcomm: The `IPQ4019` is an SoC and therefore uses distinct silicon from the other Wi-Fi 5 NICs. We could not establish silicon distinctness for the `QCA6174`/`QCA9377` pair. The `QCA9984` is a Wave 2 chipset, whereas its same-generation counterparts (`QCA6174`/`QCA9377`) are Wave 1. All other pairs follow from the generation criterion.

[4]Wi-Fi 5 has two "waves." Wave 2 adds wider bandwidth configurations and MU-MIMO, which require additional PHY functionality relative to Wave 1.

[5]Intel's proprietary architecture that keeps only the radio and signal-processing functions on the NIC and moves the remaining Wi-Fi logic to the host platform.

APPENDIX E
ARTIFACT APPENDIX

This artifact packages the components needed to build and run NICPrint, our system for identifying manufacturers and chipsets of Wi-Fi NICs. We provide the source code and compiled binaries for the full system, together with pre-trained model weights and the datasets behind the experiments we report. A complete end-to-end evaluation would require specialized hardware (e.g., a board supporting OpenWiFi and many Wi-Fi NICs) and considerable time to collect enough transmissions. Thus, we focus the artifact evaluation on computing NRAR profiles from raw pcaps and on training and testing classifiers with our existing datasets.

A. Description & Requirements

1) How to access: The Classification Module evaluated here, together with the pre-trained weights, the evaluator's guide (`README.md`) and the datasets are at https://doi.org/10.5281/zenodo.23508520. The remaining components of an end-to-end system are our `openwifi` and `openwifi-hw` forks linked from that repository.

2) Hardware dependencies: A Unix-like system is recommended: the commands assume a POSIX shell and Linux-style bind mounts, so Windows evaluators should use WSL (we provide no PowerShell equivalent). No specialized hardware is needed: experiments only require the provided datasets; hence, no GPU, SDR, or Wi-Fi NIC is required. We tested on an Apple Silicon (M2 Pro) Mac with 10 cores and 16 GB of RAM, and a x86-64 Fedora 43 machine with 20 cores and 32 GB of RAM. We recommend roughly 4 GB of disk for route (P) or 50 GB for route (R); see below for a description of each route.

3) Software dependencies: Docker Engine is the only software the evaluator installs. We tested on Docker Engine 29.7.2. If building the Docker image from source then Git is also required.

4) Benchmarks: The three dataset representations described under Experiment Workflow below, all released with the artifact. No external benchmark or additional download is required.

B. Artifact Installation & Configuration

Obtain the Docker image, either pre-built or from source:

```
# pre-built, published for arm64 and amd64:
docker pull ghcr.io/splice-project/nicprint:1.0
docker tag ghcr.io/splice-project/nicprint:1.0 \
 nicprint:1.0
# or, build it yourself:
git clone \
 https://github.com/SPLICE-project/nicprint-clf.git
docker build -t nicprint:1.0 nicprint-clf
```

Then download the dataset for the experiment to be run and create its output directories, as the `README` describes.

C. Experiment Workflow

This artifact covers the two software stages of NICPrint: the NRAR-profile computation performed by the Data-Collection Controller, and the training and inference performed by the Classification Module. We ship the dataset as raw pcaps (3.5 GB), an intermediate representation (2.2 MB), and pre-computed NRAR profiles (322 KB). Every experiment runs from either end of that chain: route (P) starts from the pre-computed data, while route (R) starts from the raw pcaps and recomputes everything, producing the files route (P) ships. Route (R) is slower but lets evaluators verify the profile computation itself; the `README` gives the download and mount commands for both. Compute times are marked as (P) or (R) in subsequent sections.

D. Major Claims

- (C1): CHIPSET-INTRINSIC IDENTIFICATION. NRAR profiles identify a Wi-Fi NIC's chipset and manufacturer on devices and wireless channels that were not part of the training set.
  - (i) *Previously unseen devices*: As shown in Section VII, average chipset-classification accuracy is 92% (RF), 99% (SVM) and 97% (LR); average manufacturer-classification accuracy is 89%, 100% and 95%, respectively. This is confirmed by E1-A.
  - (ii) *Previously unseen channels*: As shown in Section VIII, identification degrades with channel impairment, but SVM manufacturer accuracy stays above 92% (within error) at all five test positions. Moreover, manufacturer identification is consistently more robust than chipset identification. This is confirmed by E1-B.
  - (iii) *Two-stage classification* achieves a top-2 chipset accuracy at or above 90% for all three classifiers at every position except Position 2, where the target is inside an aluminum-shielded enclosure and wireless communication is severely degraded. This is confirmed by E1-C.
  - (iv) *Channel augmentation* improves accuracy for all three classifiers at every position, on both the chipset and manufacturer tasks. This is confirmed by E1-D.
- (C2): INDEPENDENCE FROM THE AUTHORS' WEIGHTS. Classifiers trained from scratch by the evaluator on the same NRAR profiles reproduce:
  - (i) the same-session cross-validation accuracies of at least 99% for manufacturer identification with all three classifiers, and 99% (RF), 96% (SVM) and 95% (LR) for chipset identification
  - (ii) the same qualitative behavior as C1: the same classes dominate the errors, and classifier rankings change only between accuracies within error of one another. Most cells fall inside E1's 95% confidence intervals.

  Both parts of the claim are confirmed by E2.
- (C3): FRAME BUDGET. NICPrint sustains high identification accuracy when the profiling cost is reduced by changing system parameters: with the number of transmitted frames per n reduced to $p \geq 250$, or the STF step size raised to $s \leq 10$, manufacturer and chipset accuracy remain above 95% for all three classifiers (both within error). This is confirmed by E3.

E. Evaluation

All experiments run inside a Docker container that can be either built from this artifact or pull from the GitHub Container Registry; the commands below are issued from the container's working directory. Outputs written to `results/` and `figs/` are visible on the evaluator's host in the corresponding `results-e*` and `figs-e*` directories (created during container's preparation as shown in the `README`).

1) Experiment (E1): [30 human-minutes + 0.1 compute-hour]: reproduces the paper's evaluations using pre-computed model weights and NRAR profiles. Supports C1.

[How to] Run the four classification scripts and compare with the tables and figures in the paper.

[Preparation] Download `nicprint-proc-data` and create the output directories as shown in the `README`, then start the container:

```
docker run --rm -it --hostname nicprint \
 --user "$(id -u):$(id -g)" --group-add 0 \
 -v "$PWD/results-e1:\
/home/evaluator/nicprint-clf/results:Z" \
 -v "$PWD/figs-e1:/home/evaluator/nicprint-clf/figs:Z" \
 -v "$HOME/nicprint-eval/nicprint-proc-data:\
/home/evaluator/DATA_ROOT:ro,Z" \
 nicprint:1.0
```

[Execution] Run the four sub-experiments in order:

```
python3 src/diff_dev.py --plot \
  | tee results/diff_dev.out
python3 src/diff_pos.py --plot \
  | tee results/diff_pos.out
python3 src/diff_pos_vote.py --top2 --plot \
  | tee results/diff_pos_vote.out
python3 src/chn_aggu.py --plot \
  | tee results/chn_aug.out
```

[Results]

(i) E1-A: `diff_dev.out`, `diff_dev.pdf` reproduces Tables VII and VIII and Figure 8: chipset accuracy 92%/99%/97% and manufacturer accuracy 89%/100%/95% for RF/SVM/LR, supporting C1(i).

(ii) E1-B: `diff_pos.out`, `diff_pos.pdf` reproduces Tables IX and X and Figures 10a and 10c; SVM manufacturer accuracy stays above 92% at all five positions while chipset accuracy degrades further, supporting C1(ii).

(iii) E1-C: `diff_pos_vote.pdf`, `diff_pos_vote_top2.pdf` reproduces Figures 11 and 12; top-2 chipset accuracy is at or above 90% at every position except Position 2, where it falls to roughly 79%, supporting C1(iii).

(iv) E1-D: `chn_aug.out`, `chn_agu_held_out.pdf` reproduces Figures 10b and 10d; accuracy improves for all three classifiers at every position relative to E1-B, supporting C1(iv).

2) Experiment (E2): [30 human-minutes + 0.5 compute-hour (P) or 6.0 compute-hour (R)]: repeats E1 with classifiers trained by the evaluator. Supports C2.

[How to] Train new classifiers on the same NRAR profiles, rerun the four E1 sub-experiments against them, and compare with E1.

[Preparation] Complete E1 first, then start the container as above with the `e2` output directories. Route (P) mounts `nicprint-proc-data` at `DATA_ROOT`. Route (R) instead mounts a writable `proc_data-e2` at `DATA_ROOT` and the raw pcaps at `RAW_DATA`, then recomputes the profiles with `utils/proc_acks.py` (see `README`).

[Execution] Train the classifiers, then repeat each E1 sub-experiment against them:

```
python3 src/train_base.py --cfm --save \
  --kind "both" \
  --models_dir ~/evaluator_models/base \
  | tee results/base_train.out
python3 src/diff_dev.py --plot \
  --models_dir ~/evaluator_models/base \
  | tee results/diff_dev.out
python3 src/diff_pos.py --plot \
  --models_dir ~/evaluator_models/base \
  | tee results/diff_pos.out
python3 src/train_base.py --cfm --save \
  --kind "special" \
  --models_dir ~/evaluator_models/spc \
  | tee results/special_train.out
python3 src/diff_pos_vote.py --top2 --plot \
  --models_mfr_dir ~/evaluator_models/base \
  --models_chip_dir ~/evaluator_models/spc \
  | tee results/diff_pos_vote.out
python3 src/chn_aggu.py --train --plot \
  --models_dir ~/evaluator_models/ch_aug \
  | tee results/chn_aug.out
```

[Results] `base_train.out` reproduces Tables III and IV: manufacturer accuracy of at least 99% for all three classifiers, and chipset accuracy of 99%/96%/95% for RF/SVM/LR, confirming C2(i). The four repeats reproduce the E1 pattern, with most accuracies inside E1's confidence intervals, confirming C2(ii).

3) Experiment (E3): [15 human-minutes + 0.2 compute-hour (P) or 5.2 compute-hour (R)]: sweeps the two profiling-cost parameters. Supports C3.

[How to] Retrain the classifiers from downsampled captures across a range of values for each parameter and inspect the accuracy curves.

[Preparation] Complete E1 first, then start the container as above with the `e3` output directories. Route (P) mounts `nicprint-ir-data` at `DATA_ROOT`. Route (R) instead mounts a writable `proc_data-e3` at `DATA_ROOT` and the raw pcaps at `RAW_DATA`, then recomputes the intermediate representation with `utils/proc_acks_ablation.py` (see `README`).

[Execution] Run the two sweeps:

```
python3 src/ablation.py --tx_num \
  | tee results/tx_num_ablation.out
python3 src/ablation.py --tx_step \
  | tee results/tx_step.out
```

[Results] `ablation_{tx,step}_*.pdf` show the accuracy curves for all three classifiers; compare these curves with the ones in Figures 13, which plot SVM accuracy only. Both sweeps meet the thresholds of C3, and accuracy degrades sharply once $s > 10$ or $p < 250$.