Countering Relay and Spoofing Attacks in the Connection Establishment Phase of Wi-Fi Systems

Naureen Hoque
naureen.hoque@mail.rit.edu
Rochester Institute of Technology
Rochester, NY, USA

Hanif Rahbari
hanif.rahbari@rit.edu
Rochester Institute of Technology
Rochester, NY, USA

ABSTRACT

To establish a secure Wi-Fi connection, a station first exchanges several unprotected management frames with an access point (AP) to eventually authenticate each other and install a pairwise key. It is, therefore, possible for an adversary to spoof elements of those unprotected frames at the physical (PHY) or MAC layers, facilitating additional attacks (e.g., man-in-the-middle and starvation attacks). Despite a few ad hoc efforts, there is still no practical way to counter these attacks jointly. In this paper, we propose practical schemes to employ cryptography at the PHY layer combined with a time-bound technique to detect and mitigate such attacks in enterprise and 802.1X-based public networks. Our backward-compatible schemes embed a digital signature of the AP (or a message authentication code) in frame preamble signals and add only a negligible delay to the connection establishment process and achieve a 98.9% true positive rate in detecting an attacker who tries to relay valid preambles. Furthermore, we conduct a formal security analysis of our scheme using a model checker and a cryptographic protocol verifier and evaluate its performance in a commercial AP-and-USRP testbed.

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**.

KEYWORDS

Wi-Fi security, formal analysis, PHY-layer authentication, MitM.

ACM Reference Format:
Naureen Hoque and Hanif Rahbari. 2023. Countering Relay and Spoofing Attacks in the Connection Establishment Phase of Wi-Fi Systems. In *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '23), May 29-June 1, 2023, Guildford, United Kingdom.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3558482.3590185

1 INTRODUCTION

The wireless local area network (WLAN) market continues to grow worldwide, with the enterprise WLAN segment reaching a record $10 billion in 2022 [1] and Wi-Fi CERTIFIED Passpoint® increasingly being adopted in dense public venues like airports, convention centers, and stadiums [2]. Even when protected by the latest WPA2 or WPA3 security protocols, however, WLANs have been the target of various forms of *multi-stage* attacks often initiated by exploiting a *pre-authentication vulnerability* [3–10]. Current Wi-Fi security protocols (including IEEE 802.11w for management frames) are able to secure frames only at the MAC layer and only after a pairwise transient key is correctly installed following a successful mutual authentication, known as four-way handshake. This leaves the frame (preamble and payload) exchanges and channel selection mechanisms prior to that point largely unprotected. A pre-authentication exploit can enable an adversary to next launch an attack to starve the users of access to idle channels [9, 10], alter data [10], decrypt (data) packets, relay or replay them, or in certain cases, retrieve the authentication key [3, 4] in its next stages.

Spoofing a frame preamble or deceiving a station into connecting to a man-in-the-middle (MitM) are two such attacks. The frame preamble is used at the physical (PHY) layer to indicate the start and duration of a frame, and a forged preamble can starve a receiver waiting for a nonexistent payload [9] or complicate frame detection [9, 10]. Offering a higher signal strength on a different channel (measured using the preamble), abusing the unprotected channel switching announcement (CSA), and jamming the channel of the real access point (AP) during the connection establishment phase are common methods an attacker can employ to launch a *multi-channel* MitM attack [3–6]. A multi-channel MitM connects to the real AP (on the original channel) on the one side, and to the station (on a new channel) on the other side, to *relay* their frames and, when advantageous, selectively delay, block, or alter management frames.

To specifically counter multi-channel MitM attacks, a MAC layer mechanism, operating channel validation (OCV), has been added to the IEEE 802.11-2020 standard to solely protect the CSA elements [11, § 12.2]. This narrowly scoped amendment was made without considering a wider range of pre-authentication attacks, e.g., preamble spoofing and even jamming-based multi-channel MitM (relay) attacks. OCV protects a specific field; it cannot authenticate the transmitter of a frame. In fact, the MAC layer is oblivious to the aforementioned spoofing and relay attacks at the PHY layer. It is therefore critical to authenticate the AP at the radio signal level to jointly counter these attacks.

In this paper, we aim to protect the connection establishment phase from relay and spoofing attacks, particularly in enterprise and IEEE 802.1X-based public Wi-Fi networks (e.g., Passpoint, eduroam), by proposing practical transmitter (AP) authentication schemes at the PHY layer. Specifically, in our digital signature scheme, a legitimate AP first signs its MAC address and a timestamp, and then tightly *chains* together all its unicast pre-authentication frames at the PHY layer by sending that signature in multiple slices–*one*

WiSec '23, May 29-June 1, 2023, Guildford, United Kingdom
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9859-6/23/05…$15.00
https://doi.org/10.1145/3558482.3590185

in each frame preamble. We can alternatively use a symmetric key approach, such as a message authentication code, and follow similar steps. Either of our designs would enable a station to cryptographically verify the transmitter of preambles in sequence (and further, the operating channel) while imposing time constraints [12] on individual frames in this chain further helps to avert relaying a valid preamble to spoof other unprotected fields of management frames.

Challenges– To introduce AP authentication for the frames in the pre-authentication phase, we need to deal with a few major challenges. **(1)** The current PHY-layer header lacks a field long enough to include a signature and certificate (or, alternatively, a message authentication code). A major PHY-layer redesign to create such a field would likely create backward-compatibility issues; besides, we aim to avoid transmitting additional frames or extending frame sizes as part of a protocol update, as that would add latency and communication overhead to the joining process. Our approach also aligns with the IEEE 802.11ai design objective of fast link setup [13]. **(2)** If an AP generates one signature/authentication code per frame and per station, it will be costly in terms of the communication and signature generation time (at the AP's end) and verification time (at the station's end). Therefore, the AP ideally needs to generate just one sufficiently strong signature/authentication code to protect the management frames. At the same time, the signature should be short enough that its slices can be communicated reliably using a preamble embedding technique. **(3)** Commercially available APs restrict modifying their firmware/baseband, where the preamble is implemented, and that limits our ability to use a testbed to fully evaluate the proposed scheme. At the same time, open-source full-stack implementations of Wi-Fi (e.g., Openwifi [14]) currently do not support the IEEE 802.11ac/ax protocol or preamble modification.

Contributions– To address these challenges, we leverage the existing trust infrastructure in enterprise and IEEE 802.1X-based public Wi-Fi networks, specifically the authentication server that itself is verified using the 802.1X framework, by requiring it to generate a public-private key pair for each AP. We then propose to use a National Institute of Standards and Technology (NIST)-approved signature that is short enough to be split and distributed across *all* unicast pre-authentication frames (sent by an AP) when using a specific communication technique, *extensible preamble modulation (eP-Mod)* [15], to embed the signature slices in the preamble signal—a technique that makes our solution backward-compatible and adds negligible delay and communication overhead each time a station needs to verify an AP. Thus, we avoid extending frame size or transmitting additional frames. To evaluate our work under a realistic setup, we first extend the *gr-ieee802-11* [16] GNURadio library to support IEEE 802.11ac (40 MHz bandwidth), then implement *eP-Mod*, and finally experimentally evaluate our proposed scheme using a commercial AP and a USRP receiver. To complement our experiments, we also formally model and analyze our proposed technique to ensure its correctness. Our main contributions are[1]:

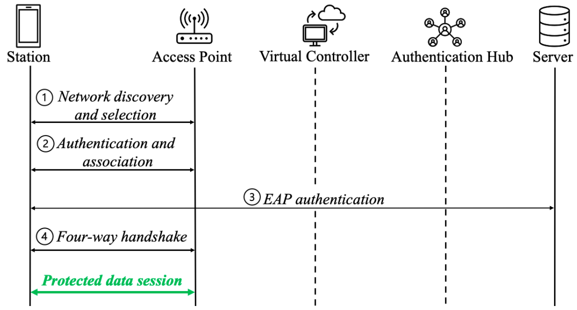


Figure 1: Wi-Fi connection establishment (virtual controller and authentication hub are present in Passpoint® networks).

(1) We devise a verification scheme to prevent an adversary from spoofing an AP's preamble, MAC address, operating channel, or device location element in the connection establishment phase of enterprise and 802.1X-based public Wi-Fi networks. We further impose time constraints on each management frame to make it harder to relay or replay legitimate preambles to spoof other frame elements.
(2) We formally model and prove the correctness of our proposed technique against various attacks using a model checker (MC), and then verify its end-to-end integrity and authenticity using a cryptographic protocol verifier (CPV).
(3) We employ supervised machine learning based on time bounds to detect simulated preamble relay attacks, achieving true and false positive rates of 98.9% and 1%, respectively.
(4) We further implement and evaluate our scheme in a commercial AP-USRP testbed to demonstrate its efficiency and practicality, showing that our technique adds only an average delay of 2.4% to the total connection establishment time.

Paper Organization– We first provide the necessary background and our system and adversary models in Sections 2 and 3, respectively. Our proposed scheme is described in Section 4. Our formal analysis is explained in Section 5. We present our simulation and experimental results in Section 6 before discussing backward compatibility and the case of Wi-Fi personal networks in Section 7. After reviewing related work in Section 8, we conclude in Section 9.

2 PRELIMINARIES

We start by briefly reviewing the Wi-Fi connection establishment phase, relevant Wi-Fi frame elements, pre-authentication relay and spoofing attacks at the PHY-layer, and preamble embedding.

2.1 Secure Connection Establishment

An enterprise Wi-Fi network consists of an authentication server, a set of APs, and possibly a large number of stations. The server is verified through a certificate authority, stores user credentials, and is responsible for authenticating users and eventually generating a pairwise master key (PMK) for each AP-station pair. The architecture and the connection establishment in Passpoint® (a.k.a. Hotspot 2.0) are similar to those of the Wi-Fi enterprise (see Figure 1). This is also true for OpenRoaming™, eduroam, and any other 802.1X-enabled public Wi-Fi networks (they are in contrast to traditional open Wi-Fi networks that offer no security at all or Enhanced Open™ that provides only unauthenticated data encryption [17]).

A station that seeks a secure Internet connection in an enterprise WLAN needs to first talk to the server via an AP.

① The first step is the network discovery and selection as shown in Figure 1. In this step, a number of (unauthenticated) management

[1] We publicly release our testbed experiments and formal verification code at https://github.com/hoquenaureen/WiFi-preauth

frames are exchanged between an AP and a station to create an initial (unsecured) connection. In active scanning mode, a station scans the network by periodically broadcasting *probe requests*. An AP may reply with a *probe response*. In passive scanning mode, an AP periodically broadcasts *beacon* frames to announce its presence to the nearby devices. When multiple APs are present, a station always prefers an AP with the highest received *signal strength* measured using those frames' training (preamble) signals [11, § 17.3.12].

② After receiving a probe response or a beacon, the station proceeds to the authentication and association steps with its preferred AP. The authentication step involves an authentication request, which is open (void) in WLAN enterprise mode, and an authentication response. Next, the station sends an association request and receives an association response. Therefore, at least three frames are exchanged in each direction in this step.

③ Once the station is connected to an AP, an extensible authentication protocol (EAP) method along with 802.1X is used in the enterprise and public modes for mutual authentication between the station and an authentication server. Some of the most commonly deployed EAP methods are EAP-PEAP and EAP-SIM. An EAP method is used to securely pass authentication information between a station and the server, where the AP (together with the controller and the hub in public networks) is a bridge between them. The AP moves to the next step only if the server confirms that the user is verified. The PMK is derived at the end of this process using a master session key (MSK), which is sent to the station via the AP using one of the *EAP* frames. Starting from the *Identity Response* frame to *EAP Success/Failure*, the AP typically transmits between 8 to 10 unique *unprotected* frames, with EAP-SIM having the least number of transmissions (8) among different EAP protocols.

④ Once the PMK is derived at both the AP and station, they perform the four-way handshake to mutually authenticate each other and derive a pairwise transient key (PTK) based on the (supposedly shared) PMK to protect the imminent (data and management) frames. A total of four EAPOL-key frames are exchanged (two from each end) in this step. They install the PTK once the handshake succeeds and start a protected session. Excluding beacons and retransmissions, the connection establishment requires an AP to transmit at least 13 and 15 unique *unprotected* management frames under EAP-SIM and EAP-PEAP, respectively.

2.2 Relevant Wi-Fi Frame Elements

2.2.1 Frame Preamble. Every Wi-Fi frame is prepended at the PHY layer by a training signal that is used by the receiver to perform certain PHY-layer functions, including frame detection, received signal strength estimation, and synchronization [11, § 17.3]. Those signals together with the SIG field (used to indicate the frame duration, among other PHY-layer information, then form a preamble.

2.2.2 Channel Switch Announcement Element. To change the operating channel in the middle of a connection establishment (e.g., when the current channel has a poor quality or has to be vacated for a radar in proximity [11, § 11.8]), an AP uses the CSA element, which can be sent within a beacon *anytime* during this phase, to advertise when it intends to switch to a specific channel. This element can also be part of an action frame (a type of management frame to already-associated stations) or a probe response.

Table 1: Recent multi-stage attacks on Wi-Fi networks that are initiated by spoofing unprotected frame elements.

Element	Attacks
Preamble	MitM [5], channel silencing [10], TaP attack [9], data alteration [10], frame detection attack [10]
CSA	KRACK [3], Dragonblood [4], FragAttack [6], multi-channel MitM [5], group key attack [7]

2.2.3 Frame Sequence Number, Timeout, Retransmission. To account for possible frame transmission failures, the standard defines a *timeout* interval for every frame and, in turn, allows retransmission of a lost or corrupted frame after that [11, § 9-10]. This interval includes transmission time, propagation and processing delays, inter-frame space and slot time, etc. Multiple retransmissions are allowed within a predefined *retry limit* until the frame is successfully received [11, § 10]. The sequence number of a frame is unique but remains constant in all retransmissions [11, § 9].

2.2.4 Device Location Element. This frame element includes the location information (latitude, longitude, altitude, etc.) and can be used by a station/AP to announce their location to others [11, § 9.4].

2.3 Spoofing/Relay Attacks in Wi-Fi

Spoofing Preambles— The preamble is not protected by the existing security protocols. Therefore, an adversary can spoof it to launch advanced multi-stage attacks—see Table 1. For example, it can force a station into silence (not able to send/receive) by sending a fake preamble but not its expected subsequent payload, effectively starving any receiver that has detected that preamble of channel access [10].

Relaying Frame Elements— In Wi-Fi, any MitM (or relay) must be a *multi-channel* one because an adversary cannot use a fake MAC address (the station can detect this during the four-way handshake) and attempting to use the same MAC address as that of a real AP on that AP's channel will be easily detected by the legitimate AP. Thus, setting up with a real AP's MAC address on a different channel is the attacker's only option (unless it uses a directional antenna to be able to relay while being on the same channel). In a CSA-based MitM, the rogue AP sends a spoofed CSA element to the station to make it switch its channel [3]. Table 1 lists a few advanced attacks that are initiated by a CSA-based MitM attack. Likewise, a jamming-based MitM involves jamming the real AP and having the station join a rogue AP on a different channel, as does offering a higher signal strength on a different channel without jamming.

Validating Operating Channel— OCV can protect CSA elements [11, § 12.2] by mandating an authenticated operating channel information (OCI) element in each frame, preventing CSA-based MitM attacks [18]. This technique adds 7% latency to the existing connection establishment because of the extra frames exchanged in each channel switch [19]. Also, this technique cannot prevent a non-CSA MitM (relay) attack or a preamble-based spoofing one.

2.4 Embedding Bits in the Preamble Signal

Using the eP-Mod technique [15], user-defined bits can be embedded in the preamble[2] in a backward-compatible way. Specifically, the preamble signal in 802.11ac Wi-Fi systems is shown to be able to

[2] Henceforth, *preamble* refers specifically to the training signal in this paper.

reliably contain up to 20 bits per frame with a 40 MHz channel and achieve the same or even better bit error rate (BER) performance of the BPSK modulation scheme [15]. More bits can be embedded with more spatial multiplexing or with less stringent BER performance even under noisy channel conditions, as extensively studied in [15].

3 SYSTEM & ADVERSARY MODEL

System Model—We consider an IEEE 802.11ac/ax Wi-Fi network configured with WPA3 (which enables .11w by default), 802.11w-enabled WPA2 in enterprise mode, or IEEE 802.1X-based public mode (e.g., Passpoint®, OpenRoaming™). Alternatively, we consider any Wi-Fi network that relies on a *trusted* authentication server and an EAP method to issue a valid public-private key pair for an AP and securely communicate the public key to the stations. We consider that at least three channels are available in the system. We further assume a legitimate AP with one or possibly two transmit antennas, meaning that an 802.11ac frame preamble from the AP can highly reliably embed up to 20 user-defined bits over a 40 MHz channel using the eP-Mod technique [15]. Additionally, while an AP is in a connection establishment stage with one station, it can continue broadcasting periodic beacons and connect with other stations. All stations use the beacon's timestamp or the timing synchronization function (TSF) to synchronize with the AP [11, § 9.4.1.10].

Adversary Model—We consider the de facto adversary model in network security systems, Dolev-Yao [20–22]. The adversary can eavesdrop, jam, replay, relay, and modify legitimate pre-authentication frames or their preambles, or inject new ones, but cannot decrypt the communications between the AP and the server. It has the following abilities: (i) The adversary has unlimited resources to create several fake APs with its desired MAC address(es). Both APs (real and fake) can be active at the same time, but on different channels (so the adversary evades detection). It can relay on the same channel only using a directional antenna. (ii) The rogue AP cannot be a part of the trusted server's network (i.e., not an insider). Also, it cannot physically tamper with a real AP (or station). (iii) The adversary does not have any access to the real AP's private key. Likewise, the user credentials (identity and password) of a station are not available to the adversary. (iv) The embedded preamble bits are visible to the adversary because they are not encrypted. Having said that, the adversary does not have any access/knowledge of the preamble bits of a frame that an AP has not transmitted yet.

Goal—The adversary's main goal is to relay, alter, or spoof pre-authentication management frame(s) or signals at the PHY/MAC layers to launch an attack (e.g., starvation or multi-channel MitM).

4 PROPOSED VERIFICATION TECHNIQUES

We propose that the trusted authentication server in enterprise/ Passpoint WLANs provides a station, at the time of each connection establishment, with either the symmetric or the public key of the AP. Our verification design can accommodate each of these alternatives with different levels of scalability, granularity, and security. We start by comparing them in terms of their real-world deployment.

4.1 Symmetric vs. Asymmetric Key Approach

A symmetric key-based function, such as hash-based message authentication code (HMAC) [23], is generally faster than a digital signature in generation and verification functions. It also allows the verifier to immediately compute the same HMAC, a feature that can further enhance the security of our design (see Section 4.3). However, symmetric key-based approaches have scalability issues in practice, e.g., in large airports and shopping malls. If a single symmetric key is supposed to be shared among all the stations, then it will be likely for an adversary to obtain that key. Considering instead one symmetric key per each AP-station pair is not scalable due to the storage and key management complexity [24], especially at the authentication server. For example, the server in an international airport would need to frequently generate and maintain tens of thousands of AP-station keys while most of the stations (i.e., passengers) may not return to that airport ever again. To solve this, one might consider an expiration time for each key after which the server and the APs will delete the stored keys. However, this would still create an extra layer of key maintenance and management.

With an asymmetric key approach, however, the server would need to generate and maintain only one key pair (public and private) per AP. Although it imposes fewer storage requirements and is the better approach for large-scale (enterprise/public) networks, digital signature functions (generation and verification) are slower than those of HMAC solutions. Therefore, in scenarios in which the network size is relatively small, e.g., in an enterprise with recurring users/employees, an HMAC-based solution with key expiration will be preferred. Without loss of generality, we adopt an asymmetric approach to explain our design for both kinds of networks (enterprise and public) and discuss the advantages of using HMAC instead of digital signature only when it is relevant to our problem.

4.2 Proposed (Asymmetric Key) Scheme

To prevent spoofing and relay attacks, the AP needs to protect the pre-authentication frames it sends to a station during the connection establishment phase. To that end, we propose that an AP generates only one signature to protect all unprotected unicast management frames it sends to a given station (we discuss alternative methods and their limitations in Section 4.4). This signature should be generated at the beginning of the process and interwoven into the PHY layer preamble. We build upon the generic eP-Mod communication technique described in Section 2.4 to embed slices of that signature in the preamble of those frames. Therefore, each frame carries one piece of the signature to protect its preamble signal, and the signature itself should be short enough that its slices can be communicated reliably (e.g., considering frame retransmissions) using the preambles. The chain of slices is further tightened using (1) the *unique sequence numbers* of a frame to detect frame insertion and preamble replay attacks; and (2) *time constraints* we impose on individual pieces of this chain to counter relaying a protected preamble when used to spoof other non-cryptographically protected content in that frame. We cryptographically protect the channel number and the sequence number of only the last frame, which helps to devise our mechanism for detecting frame insertions and tracking valid and invalid channel switches in this phase.

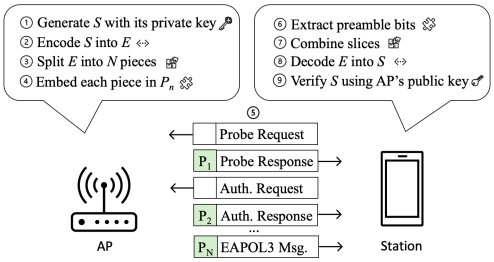


Figure 2: Proposed digital signature-based method.

We note that a symmetric-key approach can further enable checking the slice in each frame, a stronger measure to mitigate several PHY-layer attacks like preamble spoofing (see Section 4.3 for details). However, both the symmetric and asymmetric approaches should wait for the last frame before reconstructing the signature/HMAC and completing the AP verification. The steps of our proposed digital signature mechanism are described below (and in Figure 2) and Table 2 lists key notations used in this paper:

(1) Once an AP receives the first pre-authentication frame from a station, it generates a signature S over a message m using its private key. The signature may further be encoded using a channel coding scheme to increase its robustness to communication errors.
(2) The (encoded) signature, denoted by E, is sliced into N pieces, where N is the total number of pre-authentication management frames that the AP will send to the station ($N \in \{13, 14, 15\}$—see Section 2).
(3) For the last pre-authentication frame to be sent by the AP (3rd EAPOL message—EAPOL3), the final signature slice is XORed with the cryptographically hashed operating channel number and that frame's sequence number using the PTK before it embeds the bits in the preamble.
(4) Along with the MSK, the server also sends the AP's public key and device location to the station during the EAP process.
(5) Upon receiving the last signature slice, the station first verifies the frame's sequence number and AP's operating channel number. Then, after combining all the slices, the station decodes E into S and verifies the signature.
(6) The station also checks if the frames are received within the imposed time constraint t_{in} (see Section 4.3.2 for the details).
(7) If t_{in} is in the accepted range, the signature is valid, and the operating channel and the last received frame's sequence number match the cryptographically hashed ones, then the station transmits its last EAPOL message to confirm that the connection with the AP is established for a protected data session. If invalid, the station disassociates from that AP.

We now provide the design details of our proposed scheme.

Message to Sign— We choose the message $m = ID_{AP} \mid\mid t$, where ID_{AP} represents the AP's MAC address and t is the UTC time that is found in the *Time Advertisement* element of beacons [11, § 9.4][3]. As the signer by default sends the elements of m to the verifier as part of its frames, there is no need to resend m.

Table 2: Important notations.

m	Message that the AP signs
S	AP's digital signature
s	Signature size in bits
E	Encoded digital signature
N	Total pre-authentication frames sent by an AP
P_n	Embedded bits on nth frame's preamble
ChN	Operating channel number
$SeqN$	Sequence number of AP's last frame (EAPOL3)
ID_{AP}	AP's MAC address
t	UTC time of the beacon
$\mathcal{L}$	Temporal limit

AP's Public Key— The station does not need the AP's public key until it receives all of the signature slices. Therefore, instead of using any extra frames, the server sends it along with the *EAP* frame that contains the MSK. Moreover, irrespective of whether a station is joining a Wi-Fi network for the first time, it exchanges all of the pre-authentication management frames explained in Section 2. We leverage this property to eliminate the need to send the public key expiration time, as the server delivers the AP's public key to the station every time it joins the network. This completely eliminates the potential overhead from the delivery of the long chain of certificates and the key expiration time.

Digital Signature Choices. It is required to use a short signature as an AP can embed a limited number of bits in the preamble. NIST recommends maintaining a security level of at least 112 bits [25] which makes it challenging to find a short signature with a sufficient security level. In the EAP step, there can be as few as 13 frames with 20 embedded bits each; hence, the upper-bound is 260 bits. As the AP sends the signature only, not the pair (S, m), the upper limit of S stays at 260 bits. If channel coding is applied with a rate of r, then it should satisfy $\frac{s}{r} \leq 260$ bits total, or $\frac{s}{rN}$ bits per preamble.

We explore the following options for digital signature: (a) *BLS*, which is a NIST-approved signature [26] and uses a Gap-Diffie-Hellman (GDH) group and bilinear pairing for a short, but strong signature [27]. (b) *ZSS*, which is designed using the inverse computational Diffie-Hellman problem on bilinear pairings [28]. (c) *KGP*, which is an identity-based signature where the unique identity of an AP (e.g., MAC address) can be used as its public key [29]. An identify-based approach simplifies the public key distribution ahead of exchanging encrypted data. That said, in our case, the complexity for an AP to send its own public key can be reduced as the server can convey it to a station during the EAP process and we pick BLS (see Section 6). Finally, to hash the channel number and sequence number, we chose to apply Pearson's hash variant function [30].

Channel Coding— We propose applying channel coding to the signature before sending it to the station. Besides the already high reliability in *eP-Mod*, channel coding can be used optionally on the signature bits before slicing to further protect them from noise, interference, and jamming (a one-bit error will deny a connection).

Roaming— Our proposed scheme also supports seamless roaming under 802.11r, where the server shares the PMK with all the APs, skipping the EAP step and consolidating steps (1) and (4) in Section 2.1 into only two frame exchanges. In our scheme, a target AP generates an HMAC using PMK and sends any two random

[3] An AP periodically synchronizes to a UTC clock as per ITU-R Recommendation TF.460-6:2002-[B53] so that the UTC TSF offset can resolve any clock drifts [11, § 11.9].

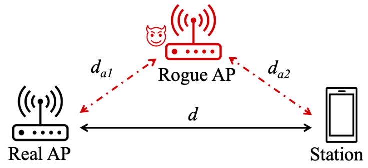


Figure 3: The adversary (relay) tries to establish a rogue AP.

slices to the station, which then locally generates the HMAC and verifies the new AP if the two slices match any of the 13 ones.

Extra EAP Frames— In the cases of an extra (unique) frame due to a specific EAP method, as soon as the AP needs to send any additional EAP frame it can pad or expand (e.g., using channel coding) the remaining signature bits among the subsequent frames. Once the station learns about the EAP method (and the extra frames), it can start decoding bits accordingly before verifying them.

4.3 Security Protocol Features

We now present a comprehensive overview of the features and security mechanisms incorporated into our proposed scheme.

4.3.1 Temporal Limit. We use the retransmission limit from the standard, shown in our earlier work to be 3 [19], as the authentication attempt limit $\mathcal{L}$. Brute-forcing the signature slices is then prevented by setting $\mathcal{L} = 3$. The probability to successfully guess *any* of the 12 slices of a particular connection (excluding the last one) within $\mathcal{L}$ is 3.4×10^{-5}. This probability is virtually zero for the last frame, EAPOL3, as the attacker would need to also successfully find a collision with the hash that is XORed with that slice (step 3).

4.3.2 Protecting Non-cryptographically Secured Frame Elements. We further protect the integrity of pre-authentication management frames as a whole, first by distributing the slices of the same signature in sequence (chaining), and then by imposing time constraints on each frame, inspired by [12, 24], to tighten the chain. Each frame must have a cryptographically valid preamble under our scheme while the slices in these preambles are designed to be mutually dependent. As a result, it would be difficult to arbitrarily spoof one slice or modify unprotected parts of a frame with a valid preamble without being detected. Therefore, in the following, we discuss why under our time-bounding scheme one cannot easily delay or alter other frame elements by relaying a valid preamble.

When an adversary captures (and possibly blocks) a frame between a station and an AP, they may try to attach a spoofed element to the payload while keeping the original preamble (i.e., the signature slice embedded in that preamble) intact, as well as the AP's MAC address, operating channel, and sequence number (if EAPOL3). The attacker could then send it to the station in an attempt to bypass the verification. The adversary would need to spoof/alter the target frame element(s) and ensure that the station receives the spoofed frame before the retransmission of the original one, which is a challenging task. (Note that the attacker cannot instead increment the sequence number to insert its own frame as the sequence number of the last frame should match the hashed one.) We impose time constraints on the inter-frame times to prevent a relay attack. We outline the inter-frame duration with and without a relay below.

Inter-frame duration when no relay (t_{sta})— Let the propagation delay be $t_{prop} = d/c$ [11], where d denotes the distance between the station and an AP, and c denotes the speed of light. In addition, there are other delays (t_{other}) involved, including transmission and processing delays, inter-frame space, and slot time. These delays (assuming transmission delay is zero) range from 0.045 – 20 ms [11, § 9]. Therefore, the inter-frame time can be estimated using $t_{sta} = t_{prop} + t_{other}$. To accurately determine t_{prop}, the station needs the *Device Location* element from a frame sent by the AP.

Inter-frame duration when a frame is being relayed (t_{adv})— Let the distance between the real AP-adversary and adversary-station be d_{a1} and d_{a2}, respectively (Figure 3). A relayed frame has to travel $d_a = d_{a1} + d_{a2}$, and $d_a \geq d$. As a result, $t_a = c/d_a$. The attacker also requires some amount of processing t_{other_a} ($\leq t_{other}$) and needs additional time t_{alter} to perform all of its adversarial actions (i.e., block a frame, copy the slice from a preamble, attach it to a spoofed frame, and then transmit). The inter-frame time that the adversary requires, $t_{adv} = t_a + t_{other_a} + t_{alter}$, must be within the duration that the station waits before it retransmits its last frame, denoted by t_{in}, i.e., $t_{adv} < t_{in}$. In Section 6, we present a machine learning technique to show that the station can reliably detect if a frame is relayed even when an adversary only relays frames without altering anything (i.e., $t_{alter} = 0$). Using a learning technique, it is possible for each device to determine its range of feasible t_{in} values based on its computational and processing capabilities. This information can be leveraged by the device as a time constraint.

An adversary may capture a probe request from a station (and jam the AP) and then send this copied probe request to the AP using the original timestamp, in an attempt to impersonate the station and create a MitM position beyond the pre-authentication phase. However, this delay in the connection establishment will ultimately detected due to the tight timing margins (t_{in}) involved.

4.3.3 Tracking (and Verifying) Channel Switch(es). A station is able to track all valid (and invalid) channel switches. If an AP is required to change the channel after sending P_n where, $n \in \{1, 2, ..., (N-1)\}$, it will send the $(n+1)$th slice over the new channel:

1. The AP sends P'_{n+1} instead of P_{n+1} over the new channel, where $P'_{n+1} = P_{n+1} \oplus P_{n+2} \oplus ... \oplus P_N$. Sending the XOR of the slices that a station has yet to receive prevents an adversary from deriving and using it to validate the channel change.
2. The rest of the frame preambles carry the regular slices: the $(n+2)$th preamble has P_{n+2}, the $(n+3)$th carries P_{n+3}, and so on.
3. Upon receiving the last frame from the AP, and before running the verification algorithm, the station will first extract the P_{n+1} by using $P_{n+2} \oplus P_{n+3} \oplus ... \oplus P_N$.
4. The station recovers the operating channel number *ChN* and last frame's sequence number *SeqN* from P_N and combine all of the signature slices.
5. If the channel change occurs after P_{N-1}, the AP will simply hash P_N with the PTK (as AP has the PTK by then). The station will first recover P_N with the PTK, then will reconstruct the full signature, and finally will verify it.

Although a cryptographically hashed channel number verification mechanism confirms that both the AP and the station are indeed on the same channel at the time of the transmission of the last pre-authentication frame (EAPOL3) by the AP, it cannot verify

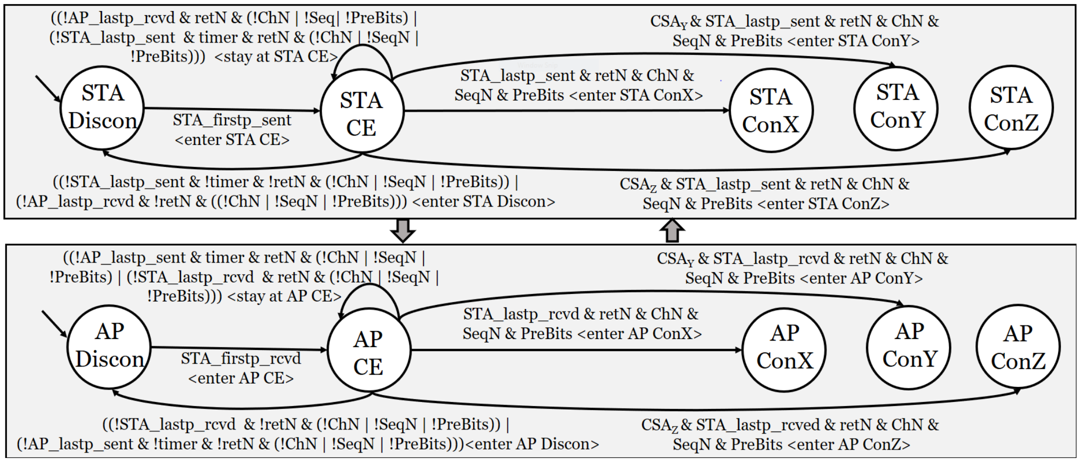

((!AP_lastp_rcvd & retN & (!ChN | !Seq| !PreBits) |
(!STA_lastp_sent & timer & retN & (!ChN | !SeqN |
!PreBits))) <stay at STA CE>
CSA$_Y$ & STA_lastp_sent & retN & ChN &
SeqN & PreBits <enter STA ConY>
STA_lastp_sent & retN & ChN &
SeqN & PreBits <enter STA ConX>
STA Discon
STA_firstp_sent
<enter STA CE>
STA CE
STA ConX
STA ConY
STA ConZ
((!STA_lastp_sent & !timer & !retN & (!ChN | !SeqN | !PreBits)) |
(!AP_lastp_rcvd & !retN & ((!ChN | !SeqN | !PreBits))) <enter STA Discon>
CSA$_Z$ & STA_lastp_sent & retN & ChN &
SeqN & PreBits <enter STA ConZ>
((!AP_lastp_sent & timer & retN & (!ChN | !SeqN |
!PreBits) | (!STA_lastp_rcvd & retN & (!ChN | !SeqN |
!PreBits))) <stay at AP CE>
CSA$_Y$ & STA_lastp_rcvd & retN & ChN &
SeqN & PreBits <enter AP ConY>
STA_lastp_rcvd & retN & ChN &
SeqN & PreBits <enter AP ConX>
AP Discon
STA_firstp_rcvd
<enter AP CE>
AP CE
AP ConX
AP ConY
AP ConZ
((!STA_lastp_rcvd & !retN & (!ChN | !SeqN | !PreBits)) |
(!AP_lastp_sent & !timer & !retN & (!ChN | !SeqN | !PreBits)))<enter AP Discon>
CSA$_Z$ & STA_lastp_rcved & retN & ChN &
SeqN & PreBits <enter AP ConZ>

Figure 4: Simplified connection establishment model with our proposed solution in place ($\mathcal{M}$); station (top) and AP (bottom).

if there have been any channel switch(es) prior to that point, including when an adversary forces invalid channel switch(es), then returns to the original one before the AP's EAPOL3 transmission. The mechanism of XORing unseen slices can detect all such valid and malicious channel switch attempts. The station can use it to track and then report to the network administrator about all valid/invalid channel switch occurrences during the connection establishment phase once it starts the protected data session. An invalid channel switch indicates the presence of an adversary, whereas, multiple valid channel switches during a single connection establishment can hint at inconsistent connections at a certain location within the enterprise or a public place (e.g., an airport).

CSA Element Reception Acknowledgement. If an associated station receives a CSA before its nth management frame transmission, then it embeds the operating channel for its next frame (P_{n+1}) in its present preamble (P_n) using the same eP-Mod technique. This acknowledgment mechanism is to prevent the AP and station from ending up on different channels during a connection establishment, similar to the Query exchange in [18]. We note that any MitM (relay) attack resulting from spoofing this acknowledgment will be detected by the station.

4.3.4 Frame-by-Frame Preamble Authentication. As discussed earlier, both symmetric and asymmetric solutions can protect the chain of pre-authentication frames. However, only a symmetric key-based approach can immediately detect preamble spoofing attacks or a corrupted preamble upon receiving *each* preamble, as both parties will have access to the symmetric key shared during the initial EAP process in order to generate the same HMAC. In contrast, under the asymmetric key-based approach, the station does not have access to the AP's private key to generate identical signature slices. When a station joins a network for the first time, however, it will not be able to use a symmetric-key approach since the symmetric key will be available only after the first EAP process.

4.4 Limitations of Alternative Approaches

One digital signature for each pre-authentication frame— If an AP generates one signature per frame, it will be costly in terms of the signature generation time (at the AP's end) and verification time (at the station's end), where the latter cannot even be performed before the station receives the AP's (public) key. Also, it will require increasing the size of every frame, which will add delay and communication overhead to the joining process and further conflicts with the IEEE 802.11ai goal for fast link setup (FILS) [13].

One digital signature of a message digest and sending it over the last frame sent by the AP— We have also considered a TLS-like solution to verify an AP's legitimacy. In TLS, application layer data across all the packets is converted into a single message digest using HMAC with a symmetric key and sent to the receiving end to protect the payload integrity during the entire process. The receiver compares it with a locally generated digest. A similar solution for our problem can be as follows. The "message" here is the payload of all pre-authentication frames sent by the AP combined. After creating the digest using a symmetric key, the AP will send it in its last pre-authentication (EAPOL3 message) frame payload. The station can verify it by comparing it against a locally generated digest. This method will not only require an extension of one frame but more importantly, cannot protect against preamble-based spoofing or multi-channel MitM attacks at the PHY and MAC layers, as discussed in Sections 1 and 2. Our aim is not only to protect the connection establishment from a multi-channel MitM, but also from the PHY-layer attacks such as starvation attack [9], frame-detection attack, and data alteration attack [10] that a TLS-like solution (merely a message digest) cannot protect. These attacks happen at the PHY or MAC layer where the upper layers are blind. Only a PHY-layer approach can prevent such attacks.

5 FORMAL SECURITY VERIFICATION

To systematically verify the correctness and security of the proposed protocol, we use a combination of model checker (MC) and cryptographic protocol verifier (CPV). To model a protocol with PHY-layer scenarios (e.g., channel switch, jamming a frame, etc.), MC is an appropriate choice, specifically for inspecting whether the model meets the temporal trace property to achieve correctness. Our protocol also uses cryptography (hash, digital signature), hence, it is important to verify its cryptographic aspects, such as message integrity, authentication of a transmitter, etc. using a CPV.

Model Checking. We create an abstract model of our proposed solution, $\mathcal{M}$, by extending the model of the existing Wi-Fi connection establishment phase we developed in our previous work [19]. The model has two finite state machines (FSMs) (one for a station and one for an AP) and each FSM is represented as a triple: (ξ, ξ_0, τ), where ξ is the finite set of states, ξ_0 is the set of initial states and $\xi_0 \in \xi$, and τ represents the set of transitions. See the simplified model in Figure 4, where, for simplicity, we only provide the main transitions between the states. Each transition in τ has two main components: condition and action. A condition specifies the logic that determines when a certain transition will occur after an action (shown in "<...>") is triggered. While an action can be null, there must always be a condition that depends on the last sent or received frame. The figure also shows different states ξ of the FSMs, where ξ_0 is the disconnected states (*STA Discon, AP Discon).* The station goes to the next state, *STA CE*, when it sends its first pre-authentication frame (*probe request*) to an AP. We assume that more than one channel switch is possible during a connection establishment phase. We introduce several new variables in this model to keep track of the channel number, sequence number, preamble bits, etc.

The adversary-controlled model, $\mathcal{M}_{adv}$, is the model that takes over $\mathcal{M}$ and performs certain action(s) based on its capabilities. We list the actions of the Dolev-Yao model-based adversary in Table 3 that we check using an MC. The MC takes $\mathcal{M}_{adv}$ as input and checks whether all possible executions of $\mathcal{M}_{adv}$ return a TRUE or FALSE. If it finds a violation (i.e., it returns FALSE), then it provides a counterexample with the traces. Each counterexample reveals a vulnerability in the system. We use the symbolic MC, NuSMV [31], to implement the model $\mathcal{M}_{adv}$, and publicly release its code. Table 3 lists the adversary actions that we checked against our proposed scheme and we find that *no* counterexample exists in the system.

Cryptographic Protocol Verification. We use ProVerif [32], a state-of-the-art automatic cryptographic protocol verifying tool, to confirm the end-to-end correctness of our proposed scheme's cryptographic security properties. ProVerif returns a FALSE if a component's verification fails. The attack model is the same Dolev-Yao model and we implement our protocol in ProVerif. Then, it runs the protocol automatically with all possible attempts by an adversary, such as trying to abuse the secrecy of an AP's private key, the integrity of the hashed message, the authenticity of the digital signature, etc. For all of the components (Table 4), ProVerif returns a TRUE (i.e., security property verified as correct) with our protocol.

6 PERFORMANCE EVALUATION

Now, we evaluate the performance of our proposed scheme using both simulations and AP-USRP testbed experiments.

Metrics– To quantify the performance of our scheme, we use the following metrics: signature generation and verification times, signature success rate (defined below), BER, and the total connection establishment time (including signature slice bit extraction time). We also measure true positive rate (TPR), false negative rate (FNR), false positive rate (FPR), receiver operating characteristic (ROC) curve, and area under ROC curve (AUC) for our relay detection technique. The signature success rate is defined as the ratio of correctly received signatures, once reassembled, to the total number of signatures. The TPR is the probability that a non-relayed frame is identified as it is, whereas the FNR is the probability that a relayed frame be identified as a non-relayed one. A ROC curve shows the TPR against the FPR. The higher the AUC (e.g., close to 1), the more accurate the relay detection is. If AUC is close to 0.5, it means that the attack detection ability is no better than random guessing. The extraction time is the time to recover one signature slice from a preamble. The total connection establishment time includes bit extraction, signature generation/verification times, and the existing Wi-Fi system's connection establishment time (which is ~ 300 ms [19]).

Table 3: Model Checker– Attacks checked.

Attack	Adversary actions
Spoofing	Adversary sends a fake CSA announcing channel switch to y to STA FSM (CSA-based MitM)
Relay	Forces AP FSM to switch channel and blocks AP's valid CSA element (jamming-based MitM)
Spoofing	Sends a fake CSA announcing channel switch to y to STA and forces AP to switch to channel z (and blocks AP's valid CSA element announcing channel switch to z) (MitM)
Spoofing	Adversary inject spoofed frame element with original preamble bits
Spoofing	Adversary injects a frame with spoofed preamble bits
Replay	Adversary replays the chain of preamble bits (i.e., replays a signature)

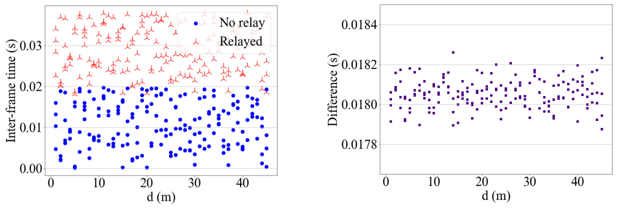
Table 4: CPV– Component verified.

Property	Automatic end-to-end verification
Secrecy	private key of AP
Integrity	AP's MAC address and timestamp
Integrity	channel and sequence number
Authentication	digital signature

6.1 Simulations

As we discussed in Section 4.3.2, an adversary needs $t_{adv} = t_a + t_{other_a} + t_{alter}$ amount of time if it wants to alter or spoof any frame element while relaying. Now we consider that it only relays the frames – it does not alter anything, then $t_{alter} = 0$. We note that it is the worst-case scenario for our detection technique. We simulate such a relay attack to evaluate the performance of the time-bound solution, considering both scenarios with and without the presence of an adversary. Our goal is to collect t_{adv} and t_{sta} and use the collected inter-frame duration to assess whether it shows any distinction between relayed and non-relayed frames.

We consider $d \in [0, 45]$ meters, which is the indoor coverage range of an 802.11ac AP [11]. For the adversary, we consider different location scenarios, where $d_a = d$ and $d_a > d$ (Figure 3). We set similar conditions for d_{a1} and d_{a2}, run the simulation 45×10^6 times, and store t_{adv} and t_{sta}. As shown in Figure 5(a), our collected values indicate that the presence of an adversary involves additional latency. The visualization of the inter-frame time differences of each sample with and without adversarial attempts (Figure 5(b)) indicates that a station can identify if a frame is relayed by looking at the inter-frame duration, as an adversary will always require an additional t_{pr} even if it is only relaying the frame. It also indicates that the time required to travel distances had no impact. To further



(a) Inter-frame times.

(b) Inter-frame time differences.

Figure 5: Data visualization of inter-frame times with and without the presence of an adversary, and their differences.

Table 5: Performance of time-bounded relay detection.

Algorithm	Accuracy	TPR	FNR
K-NN	98.3%	98.91%	1.01%
LR	98.3%	96.74%	3.21%
RF	98.0%	98.91%	1.1%

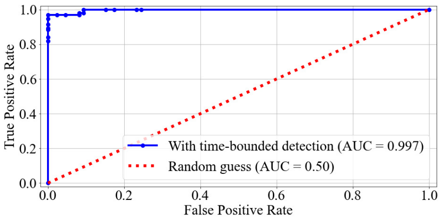


Figure 6: ROC curve and AUC of the time-bounded detection.

validate our observation that a station can identify a relay attack using the inter-frame duration, we apply machine learning techniques to our collected dataset of inter-frame duration with (t_{adv}) and without (t_{sta}) relay attack. We split this dataset into train-test parts and use *Scikit-learn* Python library to implement and evaluate K-nearest neighbor (K-NN), random forest (RF), and logistic regression (LR). During the training phase, we apply grid search to find the best parameter values (i.e., hyperparameter tuning) for optimal performance. Our results achieve 98.3% testing accuracy. Table 5 shows that only 1.01% of the time an adversary can bypass the time-bounded detection and achieve TPR of 98.91% under K-NN.

Figure 6 further shows that the AUC is 0.997, suggesting that our technique has excellent discrimination ability and can accurately distinguish between relayed and non-relayed frames. We note that the observed inter-frame delay is dependent on the particular devices, and different types of devices may display different characteristics. However, each device can pre-estimate its range of t_{in} based on its computing and processing power using a learning technique and can utilize it as a time constraint. Despite this, we emphasize that our assessment represents a worst-case scenario ($t_{alter} = 0$), and in practice, the detection technique may perform even better, especially when an adversary attempts to modify frame elements during relay attacks ($t_{alter} \neq 0$). In conclusion, our findings provide empirical evidence that supports the effectiveness of the time-bounding detection technique.

6.2 Over-the-air Experiments

Commercially available APs do not allow modifying their firmware, where the preamble is implemented, and that limits our ability to use a testbed to fully evaluate the proposed scheme. Open-source full-stack implementations of Wi-Fi such as Openwifi [14] do not support the 802.11ac protocol or preamble modification, and their FPGA implementation of the preamble is not supported in easily accessible boards either. Therefore, we extend *gr-ieee802-11* GNU-Radio library [16] to study the feasibility of our proposed method in a real setup, we use a testbed consisting of an Ettus USRP B210, connected to an Intel Core i5 host running Ubuntu 19.10 via virtual machine within the range of a real AP (Aruba)–see Figure 7.

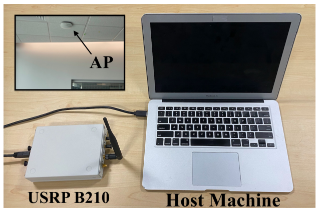


Figure 7: Experiment testbed: a static USRP B210, an Aruba AP, and a host machine in a typical research lab setting.

The *gr-ieee802-11* library supports IEEE 802.11a, not 802.11ac, but the preamble implementation is accessible within this library. Therefore, we modified it in the following way to support IEEE 802.11ac and implement the receiver-side preamble bit extraction technique: $\boxed{1}$ Since the library only supports 802.11a at 20 MHz, we expanded it to also support 802.11ac over 40 MHz of bandwidth based on the standard [11]. $\boxed{2}$ The original library discards the preamble after detecting a new frame. Since our proposed technique utilizes a sequence of preambles, we added that functionality to the library. Each received preamble then goes through the Fourier transform and channel estimation process like the rest of the frame. $\boxed{3}$ Finally, we implemented the *eP-Mod* technique as described in [15] to extract the embedded bits from the received 802.11a/ac frames. As the firmware of a commercial AP cannot be modified, we assume for simplicity that the embedded signature bits are constant.

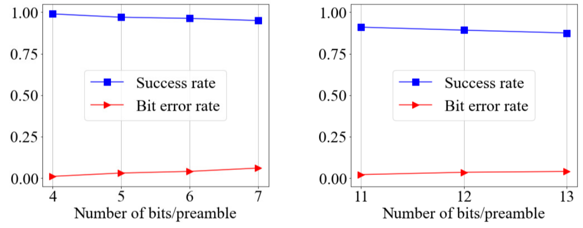
Digital signature– We measure the signature size and generation/verification times which are developed using the Pairing-Based Cryptography (PBC) library [33]. Table 6 shows that only BLS is eligible as its size is within the upper limit (see Section 4).

Success rate and BER– Since a transmitter can embed 5 – 7 bits for an equivalent BER performance of BPSK modulation in 802.11a (shown in [34]), we only explore 5 – 7 bits for each slice under 802.11a in Figure 8(a). Additionally, Figure 8(b) shows that the success rate under 802.11ac, for instance, reaches 90% when embedding 13 bits per preamble. As the 802.11ac preamble can accommodate up to 20 bits, the remaining 7 bits could be used for error correction coding, likely leading to significant enhancement in the success rate.

Extraction and total connection establishment time– No coding scheme is applied for our testbed experiments, as modifying a commercial AP's firmware is not feasible. While coding mainly enhances the BER and doesn't influence the total connection establishment time, it's worth noting that if coding could be incorporated, the success rate would be higher. Table 7 shows the number of frames needed to send a 160-bit signature for a different number of embedded bits for 802.11a/ac. In both cases, the average extraction time per frame is significantly lower than the average inter-frame duration (∼ 18.76 *ms* [19]). This indicates that shortly after each frame's arrival, a station can extract a slice from the received frame

Table 6: Signature generation/verification times with std. deviations (SD) in ms, sizes, eligibility for the proposed scheme.

Algorithm	Gen. (SD)	Ver. (SD)	Sig. Size	Eligibility
BLS	0.67 (0.08)	6.53 (0.43)	160 bits	Yes
ZSS	1.52 (0.08)	3.68 (0.12)	512 bits	No
KGP	4.46 (0.14)	4.80 (0.12)	1024 bits	No



(a) 802.11a preamble. **(b) 802.11ac preamble.**

Figure 8: AP-USRP experiments: Success rate and BER.

Table 7: Number of frames needed to send 160 bits of signature (N_{fr}), number of bits/slice ($N_{b/slc}$), the lower and the upper limits of average (T_{ax}), total bit extraction time (T_{tx}), and total connection establishment time (T_{CE}).

N_{fr}	$N_{b/slc}$	T_{ax} (ms)	T_{tx} (ms)	T_{CE} (ms)
IEEE 802.11a				
32	5	0.086	[0.086, 2.752]	[307.47, 310.13]
27	6	0.096	[0.096, 2.592]	[307.48, 309.97]
23	7	0.105	[0.105, 2.415]	[307.49, 309.80]
IEEE 802.11ac				
15	11	0.128	[0.128, 1.918]	[307.51, 309.30]
14	12	0.140	[0.140, 1.953]	[307.52, 309.33]
13	13	0.151	[0.151, 1.965]	[307.53, 309.35]

without adding any *extra* delay. Table 7 also shows the total average extraction time in the worst case (when the station starts extracting bits only after receiving all of the frames) and best case (when the station starts extracting preamble bits as soon as it receives a frame). We observe that if we increase the number of embedded bits, then the extraction takes more time. As the receiver needs to extract fewer frame preambles, in the end, the total extraction time decreases. For the case of .11ac, the number of bit variations does not have as much impact as in .11a. The total connection establishment time for 802.11a/ac is only an average of $2.4 - 3.0\%$ longer than the existing connection establishment time (300 ms as shown in [19]).

7 REMARKS

Backward Compatibility— Our proposed scheme is a backward-compatible extension to the 802.11 standard since it uses the *eP-mod* technique. *eP-mod* creates only variants of the preamble waveform that strictly satisfy the constraints necessary to support the preamble's primary functions (e.g., frame detection). This ensures backward compatibility and interoperability with legacy devices. A software update at the station is enough for the signature/HMAC verification part while stations that would not receive this update and APs that do not support our scheme can still operate perfectly.

Wi-Fi Personal Network— Our proposed solution is not effective for the personal mode as the total number of exchanged frames by an AP (6 frames) is not enough to transfer a strong signature.

Broadcast Frames— Our proposed scheme utilizes the preamble of all unicast pre-authentication frames. To protect any other element of a beacon, it can be included in the way our solution protects the channel number and frame sequence number.

8 RELATED WORK

There are different approaches to authenticate an AP's legitimacy in the literature, such as fingerprinting using carrier frequency offset (CFO) [35], phase errors between subcarriers [36], the correlation between the received signal strength (RSS) and the transmitter's location [37], etc. Changes in the transmitter hardware, channel, or configuration can affect the accuracy of such techniques, even when features are unclonable (rarely the case). Alternatively, traffic [38] or packets' time domain analysis [39] can reveal the presence of a rogue AP, but only after it is already connected to the stations.

To protect broadcast messages, timed efficient stream loss-tolerant authentication (TESLA) protocol is proposed in [40] which uses a loose time synchronization mechanism between the sender and the receivers. However, it is specific to broadcast messages and has an overhead of approximately 24 bytes for each packet. Public-key Infrastructure (PKI)- and ID-based signature solutions are proposed for the 4G/5G networks to authenticate the broadcast signals transmitted by a base station (BS) during the bootstrapping phase [24, 41]. The broadcast signals from any BS are proposed to be digitally signed with its private key where the network public key and certificates are stored in the SIM cards (a reasonable assumption in 5G cellular networks) [24]. This solution will not be feasible for a Wi-Fi network, as Wi-Fi devices do not connect to only one network (and do not have a SIM). Also, the signature generation is offline and requires storage for the pre-computed part of the signature. Therefore, it would incur high maintenance, additional storage and connection establishment time, and communication costs in Wi-Fi.

Distance-bounding protocols have been proposed in the literature to prevent relay attacks. One protocol measures the round-trip time of a challenge-response interaction between the user and verifier devices, presented in [12], while another utilizes radio frequency identification (RFID) technology to verify physical proximity, presented in [42]. In [43], the authors demonstrated the feasibility of deploying distance-bounding protocols in real-world applications. Our work is mainly centered on the investigation of the frame element modification time required by an adversary, in addition to the time taken by the frames to travel the distances between the AP and the station, as well as the (frame) processing times.

9 CONCLUSION

In this paper, we presented a practical digital signature-based scheme at the PHY layer combined with a time-bound technique to detect and mitigate relay and spoofing attacks in Wi-Fi enterprise and 802.1X-based public networks. We used a combination of formal analysis, simulations, and AP-USRP experiments to verify the efficiency and correctness of our verification scheme. We showed that our scheme can detect an adversary-relayed frame 98.91% of the time. Our evaluation under a commercial AP-USRP testbed further showed that our proposed scheme adds only an average delay of 2.4% to the total connection establishment time.

REFERENCES

[1] Worldwide enterprise WLAN growth momentum continues. https://www.idc.com/getdoc.jsp?containerId=prUS50475523. Accessed: Apr 11, 2023.

[2] Wireless Broadband Alliance. Profiles and RCOI prioritization: Improving Passpoint network selection, November 2022. Accessed: April 12, 2023.

[3] Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in WPA2. In *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, pages 1313–1328, Dallas, TX, USA, October 2017.

[4] Mathy Vanhoef and Eyal Ronen. Dragonblood: analyzing the Dragonfly handshake of WPA3 and EAP-pwd. In *Proc. IEEE Symp. Secur. Privacy (S&P)*, pages 517–533, San Francisco, CA, USA, May 2020.

[5] Mathy Vanhoef and Frank Piessens. Advanced Wi-Fi attacks using commodity hardware. In *Proc. Annu. Comput. Secur. Appl. Conf. (ACSAC)*, pages 256–265, New Orleans, LA, USA, December 2014.

[6] Mathy Vanhoef. Fragment and forge: Breaking Wi-Fi through frame aggregation and fragmentation. In *Proc. USENIX Secur. Symp.*, Virtual, August 2021.

[7] Mathy Vanhoef and Frank Piessens. Predicting, decrypting, and abusing WPA2/802.11 group keys. In *Proc. USENIX Secur. Symp.*, pages 673–688, Austin, TX, USA, August 2016.

[8] Mathy Vanhoef, Domien Schepers, and Frank Piessens. Discovering logical vulnerabilities in the Wi-Fi handshake using model-based testing. In *Proc. ACM ASIA Conf. Comput. Commun. Secur. (ASIACCS)*, pages 360–371, Abu Dhabi, UAE, April 2017.

[9] Stefan Gvozdenovic, Johannes K Becker, John Mikulskis, and David Starobinski. Truncate after preamble: PHY-based starvation attacks on IoT networks. In *Proc. ACM Conf. Secur. Privacy Wireless Mobile Netw. (WiSec)*, pages 89–98, Linz, Austria, July 2020.

[10] Zhengguang Zhang and Marwan Krunz. Preamble injection and spoofing attacks in Wi-Fi networks. In *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Madrid, Spain, December 2021.

[11] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std. IEEE 802.11. 2020.

[12] Stefan Brands and David Chaum. Distance-bounding protocols. In *Proc. Advances Cryptology (Crypto)*, pages 344–359, Lofthus, Norway, August 1994.

[13] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Fast Initial Link Setup*, IEEE Std 802.11ai. 2019.

[14] Jiao Xianjun, Liu Wei, and Mehari Michael. Open-source IEEE802.11/Wi-Fi baseband chip/FPGA design, 2019. [Online] https://github.com/open-sdr/openwifi.

[15] Zhengguang Zhang, Hanif Rahbari, and Marwan Krunz. Adaptive preamble embedding with MIMO to support user-defined functionalities in WLANs. *IEEE Trans. Mobile Comput. (TMC)*, 22(2):691–707, February 2023.

[16] Bastian Bloessl, Michele Segata, Christoph Sommer, and Falko Dressler. Performance assessment of IEEE 802.11p with an open source SDR-based prototype. *IEEE Trans. Mobile Comput. (TMC)*, 17(5):1162–1175, 2018.

[17] Discover Wi-Fi security. https://www.wi-fi.org/discover-wi-fi/security#EnhancedOpen. Accessed: June 2, 2022.

[18] Mathy Vanhoef, Nehru Bhandaru, Thomas Derham, Ido Ouzieli, and Frank Piessens. Operating channel validation: Preventing multi-channel MitM attacks against protected Wi-Fi networks. In *Proc. ACM Conf. Secur. Privacy Wireless Mobile Netw. (WiSec)*, pages 34–39, Stockholm, Sweden, June 2018.

[19] Naureen Hoque, Hanif Rahbari, and Cullen Rezendes. Systematically analyzing vulnerabilities in the connection establishment phase of Wi-Fi systems. In *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Austin, TX, USA, October 2022.

[20] Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Trans. Info. Theory*, 29(2):198–208, 1983.

[21] Syed Rafiul Hussain, Omar Chowdhury, Shagufta Mehnaz, and Elisa Bertino. LTEInspector: A systematic approach for adversarial testing of 4G LTE. In *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, San Diego, CA, USA, August 2018.

[22] Syed Rafiul Hussain, Mitziu Echeverria, Imtiaz Karim, Omar Chowdhury, and Elisa Bertino. 5GReasoner: A property-directed security and privacy analysis framework for 5G cellular network protocol. In *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, London, UK, March 2019.

[23] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In *Proc. Advances Cryptology (Crypto)*, Santa Barbara, CA, USA, August 1996.

[24] Syed Rafiul Hussain, Mitziu Echeverria, Ankush Singla, Omar Chowdhury, and Elisa Bertino. Insecure connection bootstrapping in cellular networks: the root of all evil. In *Proc. Conf. Secur. Privacy Wireless Mobile Netw. (WiSec)*, Miami, FL, USA, May 2019.

[25] Elaine Barker and Allen Roginsky. Transitioning the use of cryptographic algorithms and key lengths. https://www.nist.gov/publications/transitioning-use-cryptographic-algorithms-and-key-lengths, 2019. Accessed: January 04, 2021.

[26] Cryptographic primitives challenges and opportunities in standardization and validation of threshold cryptography. https://csrc.nist.gov/CSRC/media/Publications/nistir/8214/final/documents/nistir-8214-diff-comments-received.pdf. Accessed: April 17, 2023.

[27] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *Proc. Int. Conf. Theory Application Cryptology Info. Secur. (Asiacrypt)*, pages 514–532, Gold Coast, Australia, December 2001.

[28] Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. An efficient signature scheme from bilinear pairings and its applications. In *Proc. Int. Wrkshp. Theory Practice Public Key Crypto. (PKC)*, Singapore, March 2004.

[29] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Proc. Advances Cryptology (Crypto)*, Santa Barbara, CA, USA, August 1984.

[30] Peter K. Pearson. Fast hashing of variable-length text strings. *Commun. ACM*, 33(6):677–680, June 1990.

[31] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri. NuSMV: A new symbolic model verifier. *Int. J. Software Tools Tech. Transfer*, 2:410–425, 2000.

[32] Bruno Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *Proc. IEEE Wrkshp Comput. Secur. Found. (CSFW)*, Nova Scotia, Canada, June 2001.

[33] Ben Lynn. The pairing-based cryptography (PBC). http://crypto.stanford.edu/pbc. Accessed: March 05, 2020.

[34] Hanif Rahbari and Marwan Krunz. Exploiting frame preamble waveforms to support new physical-layer functions in OFDM-based 802.11 systems. *IEEE Trans. Wireless Commun. (TWC)*, 16(6):1545–1554, 2017.

[35] Jingyu Hua, Hongyi Sun, Zhenyu Shen, Zhiyun Qian, and Sheng Zhong. Accurate and efficient wireless device fingerprinting using channel state information. In *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pages 1700–1708, Honolulu, HI, USA, April 2018.

[36] Pengfei Liu, Panlong Yang, Wenzhan Song, Yubo Yan, and Xiuping Li. Real-time identification of rogue Wi-Fi connections using environment-independent physical features. In *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pages 190–198, Paris, France, April 2019.

[37] Yong Sheng, Keren Tan1, Guanling Chen, David Kotz, and Andrew Campbell. Detecting 802.11 MAC layer spoofing using received signal strength. In *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pages 1768–1776, Phoenix, AZ, USA, April 2008.

[38] Lanier Watkins, Raheem Beyah, and Cherita Corbett. A passive approach to rogue access point detection. In *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, pages 355–360, Washington, DC, USA, November 2007.

[39] Hao Han, Bo Sheng, Chiu C. Tan, Qun Li, and Sanglu Lu. A timing-based scheme for rogue AP detection. *IEEE Trans. Parallel Distrib. Syst.*, 22(11):1912–1925, 2011.

[40] Adrian Perrig, Ran Canetti, J.D. Tygar, and Dawn Song. The TESLA broadcast authentication protocol. *RSA CryptoBytes*, 2002.

[41] Ankush Singla, Rouzbeh Behnia, Syed Rafiul Hussain, Attila Yavuz, and Elisa Bertino. Look before you leap: Secure connection bootstrapping for 5G networks to defend against fake base-stations. In *Proc. ACM ASIA Conf. Comput. Commun. Secur. (ASIACCS)*, pages 501–515, Virtual/Hong Kong, China, June 2021.

[42] Gerhard Hancke and Markus Kuhn. An RFID distance bounding protocol. In *Int. Conf. Secur. Privacy Emerging Areas Commun. Netw. (SECURECOMM)*, pages 67–73, September 2005.

[43] Kasper Bonne Rasmussen and Srdjan Čapkun. Realization of RF distance bounding. In *Proc. USENIX Secur. Symp.*, Washington, DC, August 2010.