SecureArray: Improving WiFi Security with Fine-Grained Physical-Layer Information

Jie Xiong
Department of Computer Science
University College London
United Kingdom
j.xiong@cs.ucl.ac.uk

Kyle Jamieson
Department of Computer Science
University College London
United Kingdom
k.jamieson@cs.ucl.ac.uk

Abstract

Despite the important role that WiFi networks play in home and enterprise networks they are relatively weak from a security standpoint. With easily available directional antennas, attackers can be physically located off-site, yet compromise WiFi security protocols such as WEP, WPA, and even to some extent WPA2 through a range of exploits specific to those protocols, or simply by running dictionary and human-factors attacks on users' poorly-chosen passwords. This presents a security risk to the entire home or enterprise network. To mitigate this ongoing problem, we propose SecureArray, a system designed to operate alongside existing wireless security protocols, adding defense in depth against active attacks. SecureArray's novel signal processing techniques leverage multi-antenna access point (AP) to profile the directions at which a client's signals arrive, using this angle-of-arrival (AoA) information to construct highly sensitive signatures that with very high probability uniquely identify each client. Upon overhearing a suspicious transmission, the client and AP initiate an AoA signature-based challenge-response protocol to confirm and mitigate the threat. We also discuss how SecureArray can mitigate direct denial-of-service attacks on the latest 802.11 wireless security protocol. We have implemented SecureArray with an eight-antenna WARP hardware radio acting as the AP. Our experimental results show that in a busy office environment, SecureArray is orders of magnitude more accurate than current techniques, mitigating 100% of WiFi spoofing attack attempts while at the same time triggering false alarms on just 0.6% of legitimate traffic. Detection rate remains high when the attacker is located only five centimeters away from the legitimate client, for AP with fewer numbers of antennas and when client is mobile.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network Monitoring*

Keywords

Wireless, AoA signature, 802.11, Antenna array system, Security, SecureArray, SecureAngle

1. INTRODUCTION

WiFi networks are ubiquitous, widely deployed in both the home and enterprise, and at public hotspots. However, from a security perspective, their prevalence presents a number of well-known difficulties. Once an attacker has compromised a wireless access point (AP), she may both eavesdrop on users' traffic and inject traffic into the wireless network. Security protocols such as WPA, WPA2 (802.11i), and 802.11w have been proposed in the past few years, but they have a track record of being compromised [8, 9, 13]. Moreover, once vulnerabilities are discovered, they are slow to be fixed: Bittau *et al.* [10] report that a staggering 76% of secured APs in London had still used the older WEP security protocol six years after it was known to be insecure [31, 37]. The result is a cycle between better wireless security protocols and new protocol exploits.

Another notable trend in the design of wireless APs is the increasing number of antennas, mainly to bolster capacity with multiple-input, multiple-output (MIMO) and spatial division multiplexing techniques [6, 34, 18, 36]. Both 802.11n and the recent 802.11ac exploit MIMO extensively through the use of multiple AP antennas. Our observation is that it is possible to leverage this trend to measure the physical angles-of-arrival (AoA) at which client's transmitted signals arrive at the AP. What determines this AoA information? The multipath wireless channel between client and AP, in particular the physical propagation paths that exist between client and AP at the instant the client transmits a packet.

The dynamism of the wireless channel and the multitude of propagation paths in an indoor multipath environment present an exciting opportunity: APs can leverage this AoA information to construct an *AoA signature* that is unique to each client and extremely difficult for an attacker to forge—far more difficult to forge than previously-proposed signatures based on channel impulse response [29], received signal strength [16, 33], or other properties of the transmitter. To forge a legitimate client's AoA signature, our experiments (§4) show that an attacker's antennas would need to be co-located to within under five centimeters of the legitimate client's antennas, an unlikely event. Otherwise, even with knowledge of the legitimate client's AoA information at the AP and possession of a phased antenna array, it is still fundamentally difficult for an attacker to mimic the AoA signature, as it is highly unlikely that the set of physical paths from the attacker to the AP arrive at the same bearings as those of the legitimate client.

This paper describes the *SecureArray* system, a cross-layer approach to enhance wireless security that is designed to provide defense in depth against active attacks, those in which the attacker injects a frame into the network for the purpose of compromising the network, eliciting a denial-of-service state, or even causing a protocol deadlock and extended denial-of-service. SecureArray operates alongside and strengthens the latest protocol-based wireless security measures that establish a shared secret between the AP and legitimate client. In fact, SecureArray mitigates the most pressing current 802.11 security threats: those for which strong encryption is useless, which might be for any of the following reasons:

1. The security protocol has not yet established (in the process of establishing) a shared secret between client and AP, or
2. The 802.11 specification disallows encryption for certain protocol control messages, or
3. The shared secret has been compromised by factors outside the security protocol.

Our approach is wholly within the context of the latest 802.11 security specification for creating a "robust security network", which patches some (but not all) of the holes described above. We carefully examine the operation of the security protocol and exploit, and identify instants where the attacker must inject frames within wireless channel coherence time in order to successfully compromise the protocol. This happens with surprising frequency, often because security exploits depend on software data races with a concurrent operation at the AP that a legitimate client's transmission triggers. As an extension of this approach, we also propose *DataCheck* (§2.2.2), a new, proactive protocol that triggers a transmission within a wireless coherence time from a legitimate client when it overhears suspicious activity that may indicate ongoing spoofing.

We have implemented SecureArray with an eight-antenna Rice WARP FPGA platform acting as the AP. Our exhaustive security evaluation over one floor of a busy office space tests more than 150 sets of client and attacker locations in an active office environment with operational network traffic present in the background. Experimental results in this setting show that SecureArray achieves an overall 100% attack detection rate while simultaneously keeping the false alarm rate at 0.6% in the presence of legitimate traffic, orders of magnitude more accurate than current CSI approach [24]. Our results also show that efficacy degrades very little when we reduce the number of antennas at the AP to six and four.

The contributions of this paper are threefold: First, we propose a novel AoA signature generation scheme with random perturbations (§2.1) to increase the attack detection rate. Next, we incorporate it with, to our knowledge, the first wireless security system that uses AoA to augment existing 802.11 protocol-based security. Finally, we describe a novel proactive protocol (§2.2.2) that leverages AoA signatures to block unauthorized use of a client's security credentials, even if those credentials are compromised. We also describe solutions to two known recent 802.11 vulnerabilities, one of which results in a protocol deadlock and the other of which is a known denial-of-service issue.

In the next section, we detail SecureArray's design. Section 3 is a description of our implementation and the performance evaluation (§4) follows. The related work is in Section 5 followed by a discussion section (§6), and Section 7 concludes.

2. DESIGN

In this section we sketch the design of SecureArray, beginning with the threat model our system targets, followed by an explanation of what an AoA signature is, and a description of our AoA signature generation and comparison schemes. Next (§2.2), we describe how we integrate our AoA signature comparison algorithm with an 802.11 robust security network, the current state-of-the-art in WiFi security. It should thus become clear how SecureArray operates in concert with a security protocol to provide defense-in-depth against penetration or denial-of-service attacks.

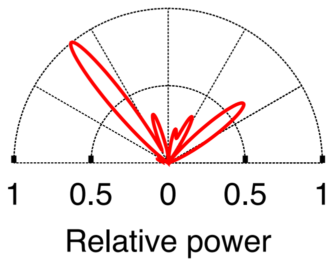

Figure 1: The *AoA spectrum* of a client's received packet at a multi-antenna AP gives an estimate of the incoming packet's power as a function of its angle of arrival.

The crux of SecureArray is to use the information contained within two or more AoA spectra resulting from two or more received packets to fingerprint the origin of each of those packets, testing whether a single client or multiple clients sent those packets. As we show in Section 2.2, knowledge of the underlying security protocol along with a method of comparing SecureArray's AoA signatures and bounds on the rate of change of the dynamic wireless channel can, with high probability, pinpoint a security breach.

Threat model. SecureArray makes no assumptions about the type of antenna that the attacker uses. In particular, the system is designed to be effective in the presence of an attacker equipped with omnidirectional antenna, directional antenna (such as the "Pringles can" antennas famously used in past high-profile WiFi security breaches [19]), or even a phased array of antennas capable of selectively beamforming and nulling transmissions to arbitrary sets of 802.11 receivers on a per-packet basis. Furthermore, our design assumes that the attacker may have breached the security protocol and may have access to the legitimate user's secret authentication credentials in the WPA2-Enterprise setup.

SecureArray offers protection only against active attacks—in particular, we do not offer any added protection against eavesdropping attacks. We do however note that in most cases to penetrate the network the attacker must transmit, and once the attacker has penetrated 802.11 protocol security, must transmit in order to send data or simply open a TCP connection.

2.1 AoA signatures

Our approach of computing AoA signatures is inspired by previous algorithms [30], but we propose an entirely novel signal processing technique in this paper to enhance the specificity of the AoA signature. In indoor environment, RF signals interact with objects in the environment, resulting in reflections, absorption, and diffraction. This results in multiple attenuated and summed copies of a client's transmitted signal arriving at an AP, a phenomenon known as wireless multipath propagation. An *AoA spectrum* of a client's received signals at a multi-antenna AP is an estimate of the incoming signals' power as a function of angle of arrival, as shown in Figure 1.

To convey the intuition of how SecureArray computes AoA spectra, we consider a single client transmitting near an AP with no multipath reflections. If the client is at a bearing θ to the AP as shown in Figure 2 (left), then its signal will travel an extra distance of $1/2\lambda \sin\theta$ to the second AP antenna, as compared with the first. This distance directly corresponds to a *measured baseband phase difference* $\Omega = \pi \sin\theta$ between the two signals' baseband symbol representations, as shown in the figure (right). Therefore, our estimate of the client's bearing to the AP $\hat{\theta}$ based on a measured

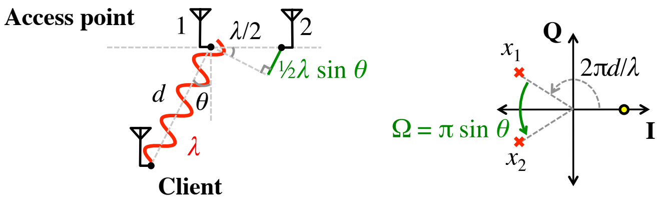

Figure 2: Principle of SecureArray's AoA spectrum computation. *Left:* The phase of the signal goes through a 2π cycle every radio wavelength λ, and the difference in distance between the client and successive antennas on the AP is governed by the client's bearing to the access point. *Right:* The complex baseband representation of the sent signal (filled dot) and received signals at the AP (crosses) reflects this relationship.

baseband phase difference Ω is

$$\hat{\theta} = \arcsin(\Omega/\pi). \quad (1)$$

In indoor multipath environments, Equation 1 breaks down, because multiple paths' signals sum in the I-Q plot. Prior work has shown that the above concepts generalize to compute AoA spectra using M antennas in the presence of indoor multipath propagation. The best known algorithm is MUSIC [30] based on eigenstructure analysis of an $M \times M$ *array correlation matrix* $\mathbf{R_{xx}}$.

Suppose D signals $s_1(t), \ldots, s_D(t)$ arrive from bearings $\theta_1, \ldots, \theta_D$ at M ($M > D$) antennas. Recalling the relationship between measured phase differences and bearing discussed above, we use the *array steering vector* $\mathbf{a}(\theta)$ to characterize the phases added relative to the first antenna, as a function of the incoming signal's bearing. For a linear array:

$$\mathbf{a}(\theta) = \exp\left(\frac{-j2\pi d}{\lambda}\right) \begin{bmatrix} 1 \\ \exp(-j\pi\lambda\cos\theta) \\ \exp(-j2\pi\lambda\cos\theta) \\ \vdots \\ \exp(-j(M-1)\pi\lambda\cos\theta) \end{bmatrix} \quad (2)$$

The array correlation matrix $\mathbf{R_{xx}}$ at AP has M eigenvalues associated respectively with M eigenvectors $\mathbf{E} = [\mathbf{e}_1\ \mathbf{e}_2\ \cdots\ \mathbf{e}_M]$. The eigenvalues are sorted in non-decreasing order, the smallest $M - D$ eigenvalues correspond to the noise while the next D eigenvalues correspond to the D incoming signals. Based on this process, the corresponding eigenvectors in $\mathbf{E}$ can be classified as noise or signal:

$$\mathbf{E} = \left[\overbrace{e_1\ \ldots\ e_{M-D}}^{\mathbf{E}_N}\ \overbrace{e_{M-D+1}\ \ldots\ e_M}^{\mathbf{E}_S} \right] \quad (3)$$

We refer to $\mathbf{E}_N$ as the *noise subspace* and $\mathbf{E}_S$ as the *signal subspace*. The MUSIC AoA spectrum is then computed as:

$$P(\theta) = \frac{1}{\mathbf{a}(\theta)^H \mathbf{E}_N \mathbf{E}_N^H \mathbf{a}(\theta)} \quad (4)$$

2.1.1 Random phase perturbation

We note that $\hat{\theta}$ and Ω do not have a linear relationship with each other. We plot estimated client bearing in Figure 3 (upper), in units of degrees and the derivation of $\hat{\theta}$ with respect to Ω in Figure 3 (lower). Here, note that a small perturbation of Ω translates to smaller perturbations of $\hat{\theta}$ when the client is broadside to the axis of the array (*i.e.*, $\theta \approx 0$), but larger perturbations of $\hat{\theta}$ when the client is close to $\theta = \pm\pi/2$ radians (corresponding to $\Omega = \pm\pi$ radians):

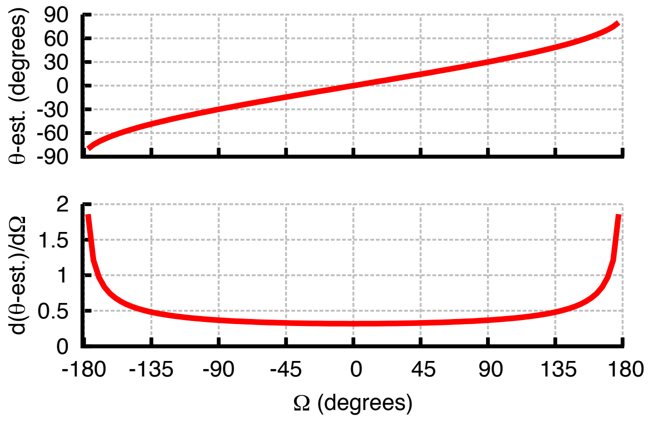

Figure 3: *Upper:* Estimated client bearing to the AP $\hat{\theta}$ as a function of measured baseband phase difference Ω, and its rate of change with respect to Ω (*lower*).

$$\frac{\mathrm{d}}{\mathrm{d}\Omega}\hat{\theta} = \frac{1}{\sqrt{\pi^2 - \Omega^2}} \quad (5)$$

we see that $\mathrm{d}\hat{\theta}/\mathrm{d}\Omega$ reaches a minimum of $\pi^{-1} \approx 0.32$ when the client is broadside, and increases sharply when the client is near the array axis. The AoA computation in Equation 4 yields one AoA spectrum. But since we are trying to formulate a highly specific client signature, we wish to make the signature sensitive to slight changes at any angle. Based on the above observation of non-uniform rate of change of $\hat{\theta}$ with respect to Ω, we propose a novel scheme to add a random phase offset ζ_2 to Ω, and compute another AoA signature. We iterate this process, adding $L - 1$ further random offsets[1], so obtaining L AoA signatures $\sigma_1(\theta), ..., \sigma_L(\theta)$ based on L random phase offsets $\zeta_1 = 0, \zeta_2, \ldots, \zeta_L$.

Note that we could have introduced a deterministic perturbation to bring the bearing θ close to $\pm\pi/2$ to make the signature most unique in order to increase the attack detection rate. However, deterministic perturbation has the following disadvantages compared with random perturbations:
1. The false alarm rate will be increased accordingly.
2. The bearings very close to $\pm\pi/2$ become unstable (sensitive to very tiny changes).
3. As there are multiple peaks on a signature, it's difficult to bring all the bearings near to $\pm\pi/2$ at the same time.

SecureArray thus employs random perturbations to spread the bearings all over the range between $-\pi/2$ and $+\pi/2$ to maintain a balance between high detection rate and low false alarm rate. The averaging process described in next section mitigates the inaccuracies caused near $\pm\pi/2$ radians and also reduces the possibility of similar peaks in coincidence.

We demonstrate how random perturbation works with Figure 4. Figure 4 (upper) shows the effect of random perturbation on two frames transmitted 100 milliseconds apart, from a legitimate client and an attacker placed five centimeter away. The signatures are similar (but not identical) under one random phase perturbation, with a rather high similarity metric $\mathcal{M} = 0.79$ (§2.1.2), but the other random phase perturbation produces relatively different sig-

[1]For more than two antennas, each offset here is an offset vector with size $M - 1$ while M is the number of antennas at each AP.

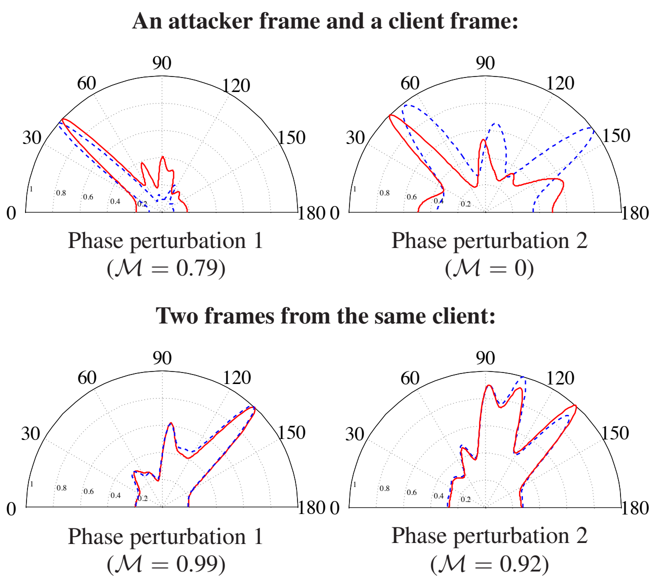

Figure 4: The effect of random phase perturbation on AOA signatures: *Upper:* comparison between an attacker's frame and a client's frame. *Lower:* comparison between two frames from the same client. $\mathcal{M}$ is the similarity metric (§2.1.2).

natures with a similarity metric $\mathcal{M} = 0$. So with multiple random random perturbations, we have a higher chance that the two signatures are separated with some of the perturbations introduced. Figure 4 (lower) shows that the random perturbations do not separate (which is desired) the AoA signatures of frames (also spaced 100 ms apart) from the same client, enhancing the AoA signature specificity and selectivity.

2.1.2 *AoA signature comparison algorithm*

Given two AoA signatures $\sigma^{\mathrm{A}}(\theta)$ and $\sigma^{\mathrm{B}}(\theta)$, we first find local maxima in each signature using standard numerical methods. We design a metric $\mathcal{M}$ that pairs local maximum i from σ^{A} with local maximum j from σ^{B}. The metric takes two pseudospectra σ^{A} and σ^{B} as inputs, and pairs peaks only if they are positioned within a small constant angle threshold Θ of each other. The similarity metric $\mathcal{M}$ also takes the magnitudes (normalized) of paired peaks into consideration. If two peaks are paired in coincidence, their peak magnitudes may vary significantly. In order for $\mathcal{M}$ to approach one, the peaks need to be paired and at the same time, the magnitudes of paired peaks should also be close to each other.

$$\begin{aligned} S &= \{(i,j) : |\angle i - \angle j| < \Theta\} \\ \mathcal{M}\left(\sigma^{\mathrm{A}}, \sigma^{\mathrm{B}}\right) &= \frac{\sum_{(i,j)\in S} m_i \cdot m_j}{\left(\sum_i m_i^2 + \sum_j m_j^2\right)/2}. \end{aligned} \tag{6}$$

To incorporate the random phase perturbation scheme described in Section 2.1.1, we obtain one similarity metric with one random phase perturbation by comparing pseudospectra $\sigma_1^{\mathrm{A}}(\theta)$ and $\sigma_1^{\mathrm{B}}(\theta)$. We iterate this process to obtain L metrics from comparing the L pseudospectra $\sigma_1^{\mathrm{A}}(\theta), ..., \sigma_L^{\mathrm{A}}(\theta)$ arising from random phase perturbations of σ^{A} pairwise with the L pseudospectra $\sigma_1^{\mathrm{B}}(\theta), ..., \sigma_L^{\mathrm{B}}(\theta)$ arising from σ^{B} with the same random phase perturbations applied. We average the L metrics obtained to achieve $\bar{\mathcal{M}}$.

Based on empirical data from our performance evaluation below, we use a binary classification threshold test, choosing a threshold η, classifying signature pairs (σ_1, σ_2) as different (thus flagging an attack) if $\bar{\mathcal{M}} < \eta$, and same (thus identifying normal traffic) if $\bar{\mathcal{M}} \geq \eta$. Table 2 (page. 7) defines the full confusion matrix for the terms we use in our evaluation, and we report our choice of η in Section 4.2.1.

Speed	Coherence time at 2.4 GHz
Near-stationary (1 kph)	77 ms
Walking (5 kph)	15 ms
Running (12 kph)	6 ms

Table 1: Wireless coherence times at 2.4 GHz as computed with Equation 7 for typical client motion speeds.

Wireless coherence time. The wireless channel between client and AP is determined by scattering, reflection, and refraction by objects in the environment. A key design parameter is the time duration over which the wireless channel can be considered unchanging with high likelihood, as a function of the speed of the client's motion and/or motion of objects in the environment. The wireless *coherence time* T_c is determined by carrier wavelength λ and maximum client velocity v. If we measure v in meters per second, then the coherence time is given by [35]:

$$T_c = \frac{9}{16\pi(v/\lambda)}. \tag{7}$$

Table 1 shows wireless coherence time at 2.4 GHz as a function of typical client motion speeds indoors. Within this time, we can be confident that the AoA signatures of two transmissions from the same client will indeed be the same, making a false alarm (see Table 2) a rare event. Note that SecureArray introduces very little amount of overhead, as SecureArray is not triggered all the time but activated only when some particular control packets are received or unusual behaviors are detected.

2.1.3 *Collision detection and attacker jamming*

In many cases an attacker will have no incentive to jam a client's uplink transmission, but in some cases she may have an incentive to do so. The attacker can employ two directional antennas: one pointed towards the legitimate client and the other pointed at the AP. The attacker uses one directional antenna to jam the client's transmission at the AP and records the client's transmission with another directional antenna. The attacker then replays the recorded packet to cheat the AP. We will illustrate this jamming and reply attack in more detail in Section 2.2.4. SecureArray performs collision detection by searching for energy changes and preambles in the middle of incoming frames as is a standard practice in prior work [17, 36]. When the attacker jams the legitimate client's data frame, the AoA for the overlapping portions of the two packets will be the superposition of the two signatures σ_1 and σ_2 as shown in Figure 5, which SecureArray's classification algorithm will flag as differing from the client's signature σ_1.

2.2 Robust security network integration

We now explain how SecureArray can integrate with an 802.11 robust security network, mitigating attacks against the security protocol itself as well as mitigating authenticated but unauthorized spoofing of a user's stolen credentials. To integrate SecureArray with the entirety of the 802.11 protocol, we envision AP designers will follow similar reasoning, annotating their implementations with packet pairs or triplets in the security protocol exchange between client and AP where the AP should make an AoA signature comparison, and a proper corresponding action to be taken if that comparison fails.

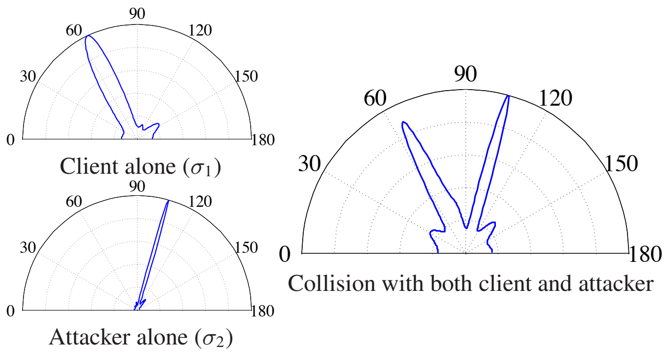

Figure 5: The result of an attempted jamming attack on DataCheck: the attacker's jamming attempt *(right)* is discernible as the superposition of its own AoA signature atop the legitimate client's AoA signature *(left up)*.

2.2.1 Deauthentication deadlock attack

WPA2 uses IEEE 802.1X and in turn the Extensible Authentication Protocol (EAP) in order to authenticate wireless clients to an authentication server. EAP messages defined by 802.1X are referred to as *EAP over LANs* (EAPOL). Recently, Eian and Mjølsnes [14] and Bertka [9] have separately observed that an attacker can inject an unauthenticated deauthentication notification after the third message of the EAPOL four-way handshake (see Figure 6) that takes place when a client connects to a WPA2 AP. Furthermore, since the attack takes place during the EAPOL four-way handshake, amendments to 802.11 (802.11w [2]) that protect management frames (and in particular deauthentication frames) do not mitigate this vulnerability [9]. A successful attack causes a deadlock, leading to denial of service for the wireless client. Note that for the attack to be successful, the attacker must inject the deauthentication packet between the client's receipt of EAPOL message three and the client's transmission of EAPOL message four.

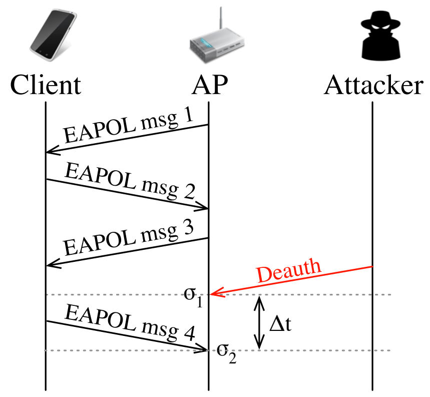

Figure 6: *EAPOL Deauthentication deadlock attack.* The attacker sends a "deauth" message to a client whose 802.1x four-way authentication handshake is in progress. The AP makes an AoA signature comparison between the attacker's deauth packet (σ_1) and the legitimate client's subsequent EAPOL message four (σ_2).

Mitigating deauthentication deadlock. Using a wireless sniffer operating in monitor mode, we measured typical interpacket spacing times for 100 EAPOL four-way handshakes on a variety of AP platforms. We find that the vulnerability gap (Δt in Figure 6) between EAPOL message three and EAPOL message four ranges between 30–59 μs with an average value of 36 μs, well within even a running-speed wireless coherence time (*cf.* Table 1), enabling the AP to make a rapid AoA signature comparison (§2.1.2) between the attacker's deauthentication packet and the legitimate client's subsequent EAPOL message 4, flagging an attack and dropping the attacker's frame at the link layer if the signatures do not match.

2.2.2 Authenticated spoofing attack

In many cases, users select easily-guessable passwords that are susceptible to dictionary or social engineering attacks, and the attacker gains access to the network using an authorized user's WPA2-Enterprise login credentials. By thus authenticating and associating with the AP, the attacker can sniff the user's MAC address and then inject packets that are completely identical to the legitimate user via the AP.

SecureArray can mitigate these types of attacks when clients overhear acknowledgment frames resulting from injected traffic. We now present *DataCheck*, a protocol that uses this mechanism in conjunction with AoA signatures to mitigate this traffic injection.

DataCheck: Mitigating authenticated spoofing. We propose a client-assisted protocol to mitigate authenticated spoofing that requires minimal protocol changes to 802.11 clients, focusing instead on the addition of AoA signature processing at the AP.

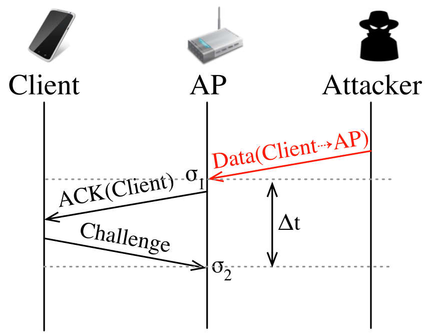

Figure 7: *Authenticated spoofing attack and mitigation with DataCheck.* Clients send protected minimal-length data frames or normal traffic in response to unexpected ACKs from the AP. The AP makes an AoA signature comparison between the attacker's data frame (σ_1) and the legitimate client's subsequent *challenge* data frame (σ_2).

Figure 7 outlines the protocol. As part of the 802.11 Distributed Coordination Function (DCF) medium access control, clients normally maintain a state as to whether they are awaiting an ACK from a frame they sent to the AP. DataCheck modifies the DCF so that if a client receives an ACK for a frame that it did not send (or even overhear, in the case that the attacker uses a phased array to direct the frame solely to the AP and null the client) the client sends a *challenge* (another data frame, or a zero-length data frame if it has no data outstanding) to the AP. The AP then makes a signature comparison between the attacker's data frame (σ_1) and the legitimate client's challenge frame (σ_2) and flags the exchange as suspicious if the two signatures do not match **and** both data frames are protected with 802.11 security.

If the attacker has the legitimate user's security credentials, she could trigger an alarm by injecting a (security-protected) data frame after overhearing a legitimate data-ACK exchange between client and AP, but this simply alerts network administrators that the user's account has been compromised. If the attacker doesn't have the user's security credentials, the encryption check on the DataCheck data frame (σ_2) prevents her from triggering a false alarm.

Through the application of AoA signatures and in the presence of a saturated wireless network, DataCheck incurs no additional overhead on the wireless medium. It also does not require any form of header format changes.

2.2.3 *Authentication deadlock attack*

In 2011, Eian *et al.* used model checking to uncover another distinct authentication-related denial-of-service vulnerability in 802.11i [13]. In this case the vulnerability appears to be based on implementation flaws in hostapd's implementation of 802.11i. As shown in Figure 8, the attack is possible when the AP and client are taking turns sending data frames to each other, each decrypting data frame as they arrive from the other. Just after the attacker overhears a data from the client to the AP, she injects an Open System authentication request frame for the same client to the AP. This causes the AP to delete its security association with the station, but internally, its security state machine transitions to the wrong state (possibly because of a race condition). Since the AP has deleted its key for the victim client, it cannot transmit or receive any frames for that client, and since the AP is in the wrong state, resynchronization mechanisms that exist in the 802.11-2012 standard are not triggered. The result, therefore, is a deadlock that typically results in client disconnection for periods of several minutes.

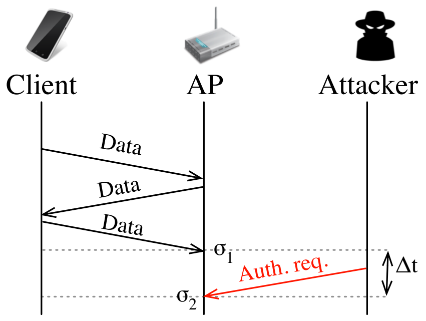

Figure 8: ***Authentication deadlock attack.*** **The attacker sends an authorization request message to the AP just after it has received and is decrypting a data frame from a legitimate client. This results in loss of authentication state and a subsequent protocol deadlock.**

SecureArray is ideally suited for detecting and mitigating attacks such as these because the time period over which the AP is vulnerable Δt is the immediate few hundred microseconds after it receives a frame from the legitimate client. Indeed, the attacker needs to inject the frame in an early backoff slot after just a DIFS time spacing (from 34 μs in 802.11a to 50 μs 802.11b and 802.11g networks backwards compatible with 802.11b) in order to win the 802.11 MAC protocol contention. This is clearly within the coherence time, making a robust AoA signature comparison possible.

Mitigating authentication deadlock. If SecureArray finds a mismatch between the AoA signature of the authentication request frame σ_2 and the AoA signature of the data frame σ_1, it drops the spurious authentication request frame. Note that since the authentication request must nominally come from the legitimate client already authenticated in order for the attack to be effective, this policy will not affect other legitimate clients attempting to authenticate in the presence of heavy traffic.

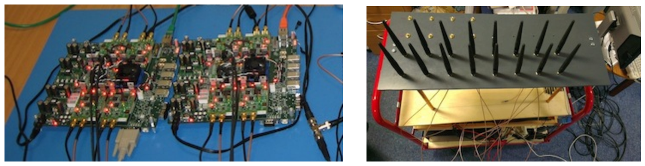

Figure 9: ***Left:*** **the SecureArray prototype AP is composed of two WARP radios, while a cable-connected USRP2 software-defined radio (not shown) calibrates the array.** ***Right:*** **The AP and antenna array mounted on a cart.**

2.2.4 *Jamming and replay attack*

Advanced attack such as *jamming and replay attack* can be initiated by attacker with two directional antennas. Take deauthentication deadlock attack as an example shown in Figure 6, the attacker jams the reception of EAPOL message four at the AP with one directional antenna, while at the same time records message four with another directional antenna. The attacker then replays the recorded message four to the AP so both deauthentication packet and message four are now from the attacker which means the AoA signatures are the same. However, when the attacker jams the reception of message four, the AoA at the AP during jamming is a superposition of bearings from both the legitimate client and attacker. This causes the AoA signature to vary significantly as shown in Figure 5, which can be easily detected.

3. IMPLEMENTATION

SecureArray AP. The prototype SecureArray AP, shown in Figure 9, uses two Rice WARP FPGA-based platforms. Each WARP has four radio front ends and four omnidirectional antennas. Digital I/O pins on one of the two WARPs output a time synchronization signal on a wire connected between the two, so that the second WARP board records and buffers the same time-indexed samples as the first. The WARPs run a custom FPGA hardware design architected with Xilinx System Generator for DSP that implements the functionality described above (§2).

We place the eight antennas attached to the WARP radios in a linear geometry (Figure 9, right). Antennas are spaced at a half wavelength distance (6.13 cm) to yield a 180 degree spectrum range. This also happens to yield maximum MIMO wireless capacity, and so is the arrangement preferred in commodity APs.

AP phase calibration. Equipping the AP with multiple antennas is necessary for SecureArray. Note that calibration is mandatory for AoA localization purpose as described in [39] but optional for SecureArray. As the random perturbation AoA scheme is employed, SecureArray is immune to random phase offsets introduced by oscillators and hardware imperfections. However, as calibrated version of AoA has the true signal arrival bearing information, the peak corresponds to the direct path signal is relatively stable when the client is mobile, including this calibrated version of AoA into SecureArray reduces the false alarm rate. So we still describe the calibration scheme employed for SecureArray:

Random phase offsets introduced by WARP: Each radio receiver has 2.4 GHz oscillator whose purpose is to *downconvert* the incoming radio frequency signal to its representation in I-Q space. This

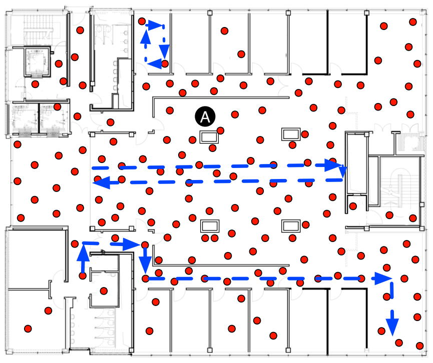

Figure 10: Floor plan of our evaluation environment. Mobile experiment trajectories are shown by the dashed lines.

introduces an unknown phase offset to the resulting signal. These unknown phase offsets are constant during each WARP power on/off cycle which only need to be calibrated once.

Hardware imperfections: Because of hardware imperfections for SMA splitters and small differences between cables of the same nominal length, we use the following one-time (run only once for a particular set of hardware) calibration scheme to handle these equipment imperfections.

To remove the hardware imperfections and obtain the unknown random phase offsets introduced by oscillators, we calibrate the array with a USRP2 generating a continuous wave tone. The signal from the USRP2 travels through splitters and cables (we call them *external* paths, giving rise to an external phase difference) before reaching the WARP radios. The phase offset ϕ^I we want to measure is the *internal* (due to RF downconversion) phase difference between radios one and two, $\phi_2^I - \phi_1^I$. Running cable calibration once, we obtain offset ϕ_1:

$$\phi_1 = (\phi_2^E + \phi_2^I) - \left(\phi_1^E + \phi_1^I\right) \tag{8}$$

Because of equipment imperfections, ϕ_2^E is slightly different from ϕ_1^E, so ϕ_1 does not equal the quantity we want to measure, ϕ^I. We then physically exchange the external cable connections and re-run cable calibration:

$$\phi_2 = \left(\phi_1^E + \phi_2^I\right) - \left(\phi_2^E + \phi_1^I\right) \tag{9}$$

Combining Equations 8 and 9, we obtain the desired quantity:

$$\phi^I = \frac{1}{2}(\phi_2 + \phi_1) \tag{10}$$

Testbed clients. The clients we use in our experiments are Soekris boxes equipped with Atheros 802.11g radios operating in the 2.4 GHz band.

4. EVALUATION

We first describe our methodology (§4.1), then evaluate SecureArray's performance when clients are static (§4.2), followed by an evaluation under mobility (§4.3). Finally, we examine system latency of SecureArray (§4.4).

4.1 Methodology

For static clients, we deploy them at 150 randomly-chosen positions (and then deploy the attacker randomly at a particular distance with respect to the client) in a 30 × 40 meter typical office environment, as shown in Figure 10. We aim to have both line-of-sight (LOS) and non-line-of-sight (NLOS) clients in our test environment. We also deliberately place the client and attacker behind and near to human, ceramic, wood, plastic and metal materials in the office to make our results more comprehensive and realistic. We also run the experiments in both daytime and night time. We evaluate SecureArray with signature comparisons from packets recorded with different time gaps from one millisecond to two seconds, and also vary the distances between the client and attacker from above three meters down to just five centimeters to challenge our system.

For mobile clients, we evaluate the challenging scenarios when both the client and attacker are moving at the same time. The experimenter holds the client and attacker in his two hands walking smoothly following the predefined routes. We evaluate SecureArray with signatures from packets recorded with different time gaps.

		Test: Do signatures differ?	
		Yes	**No**
Ground truth: Are signatures from different senders?	**Yes**	TP: *Attack detected*	FN: *Missed attack*
	No	FP: *False alarm*	TN: *Normal traffic*

Table 2: Confusion matrix defining the terms SecureArray uses to classify signatures. *TP:* true positive, *FN:* false negative, *FP:* false positive, *TN:* true negative.

4.2 Static experiments

We first show how SecureArray performs in a typical office environment when both client and attacker are static. We evaluate SecureArray with all combinations of different locations, different distances between attacker and client, and different time intervals between the packets, which is a total of 1500 comparisons.

4.2.1 Overall system performance

As shown in Section 2.1.2, the similarity metric $\mathcal{M}$ obtained varies between zero and one. This metric should be compared with a carefully chosen threshold η to reject packets from attacker and accept packets from legitimate client. Table 2 shows the confusion matrix defining the terms that we use to classify the signature comparison test described in Section 2.1.2. To choose the best threshold η, we examine the *receiver operating characteristic* (ROC) curve. This plots the attack detection rate $\frac{\text{TP}}{\text{TP+FN}}$ versus the rate of false alarms $\frac{\text{FP}}{\text{FP+TN}}$ on signature comparisons from the same sender, for varying choices of threshold η.

We show the overall ROC curves in Figure 11, averaging across different packet time gaps, and different distances for maximum generality. Besides SecureArray, we also present the curves for a *phase-calibrated* version (remove the phases introduced by WARP and hardware imperfections), and a *random phase* version (adding one group of eight random phases to each of the radio board). We see that SecureArray achieves a 100% attack detection rate at a false alarm rate as low as 0.67% with L set to 15. We see clearly that SecureArray outperforms the two other versions with $L = 1$.

Choice of η. To maintain a balance between attack detection rate and false alarm rate, we choose $\eta = 0.7$ and evaluate with this threshold in the remainder of the evaluation.

4.2.2 Number of random-phase perturbations (L)

We show the effect of number of groups of random perturbations employed on SecureArray here. As shown in Figure 12 and the zoomed in version in Figure 13, with more groups of random per-

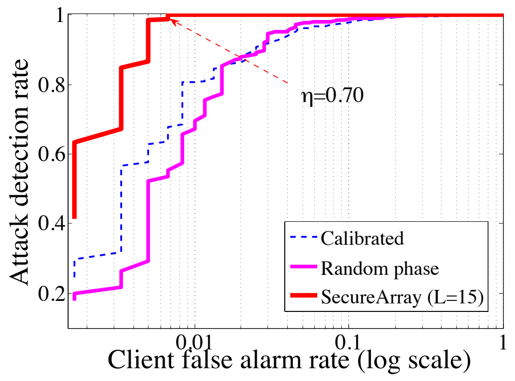

Figure 11: Overall performance (*receiver operating characteristic* (ROC) curve) using calibrated AoA signatures, AoA signatures perturbed by one random phase, and SecureArray (with $L = 15$).

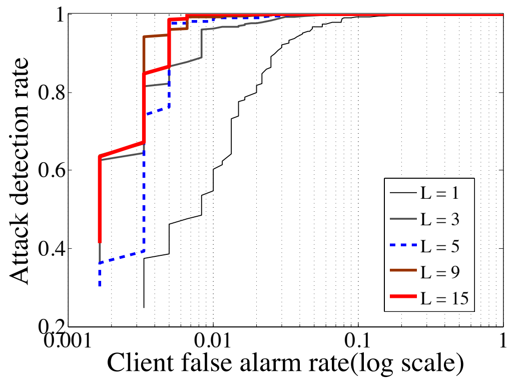

Figure 12: ROC curves for SecureArray, varying the number of random phase perturbations attempted for each AoA signature from $L = 1$ to $L = 15$.

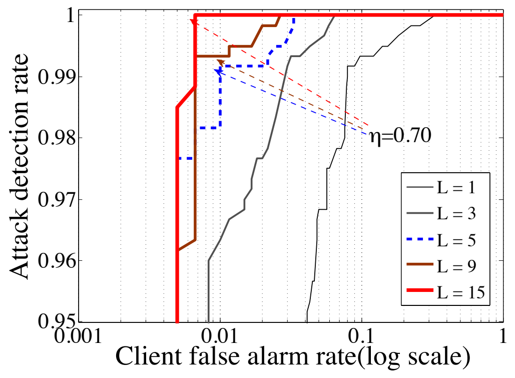

Figure 13: Detailed view of the ROC curves in Figure 12, for high attack detection rates.

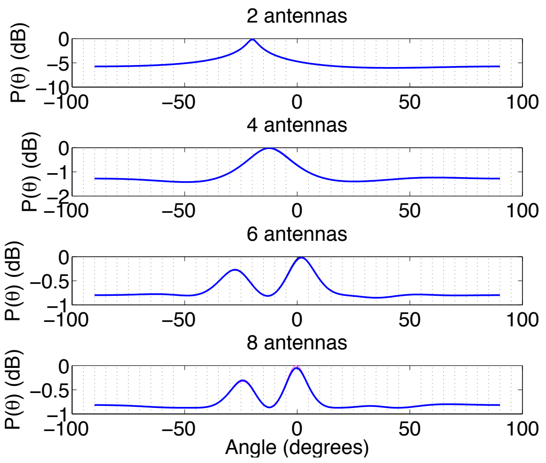

Figure 14: Effect of the number of antennas on the AoA signature: more antennas yield more unique signatures.

turbations employed, performance improves further. The biggest improvement gap is between $L = 1$ and $L = 3$ which demonstrates the efficiency and necessity of our random phase perturbation scheme. However, this improvement becomes insignificant when we increase L past five. We should notice that this improvement is achieved at the price of more computation with a larger L. However, as shown later in Section 4.4, the computational load involved for SecureArray is very little and a typical desktop can complete the process within milliseconds. Furthermore, the L computational processes are independent from each other which can be executed in parallel at the same time. For all subsequent results presented, if not mentioned, we employ $L = 5$ groups of randomly perturbations for averaging.

4.2.3 *Number of AP antennas*

The number of antennas at the AP is an important factor for computing the AoA signature and accordingly for SecureArray performance as it decides the maximum number of lobes on the signature and the sharpness of the lobes. With eight antennas, we can have a maximum of seven lobes on the signatures corresponding to seven incoming signals (two signals arriving at very close bearings will merge into one lobe on the signature). With more lobes, the signature has a tendency to be more specific to each client and easier to be differentiated, as shown in Figure 14. We present the ROC curves for SecureArray with varying numbers of antennas in Figure 15. With a threshold $\eta = 0.7$ and $L = 5$, SecureArray achieves 99.2%, 99.8% and 99.3% attack detection rate and 1%, 4.7% and 11.3% false alarm rate respectively.

We observe that with our chosen threshold of $\eta = 0.7$, the attack detection rate remains high even with four antennas. However, the client false alarm rate does increase a little bit with fewer numbers of antennas. As the next generation 802.11ac supports up to eight spatial streams, eight antennas are very likely to be equipped on one single AP in the near future.

4.2.4 *Distance between client and attacker*

In most scenarios, an attacker will be far away from the legitimate client which is the scenario most prior work [24] evaluates. However, with devices becoming smaller, it's plausible that the attacker may be very close to the legitimate client, such as when a smartphone is placed in the same bag or on the same table. We would like

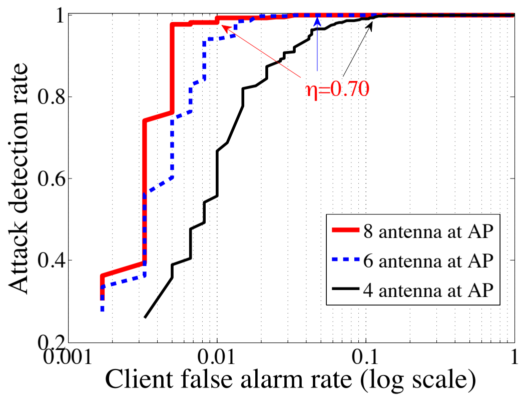

Figure 15: Performance of SecureArray with different numbers of antennas at the AP.

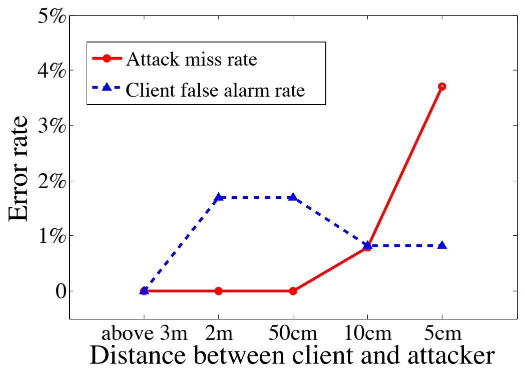

Figure 16: Performance of SecureArray with varying distances between client and attacker.

to evaluate SecureArray's performance for these most challenging scenarios when the attacker is very close to legitimate client.

In this section, we present results for different distances between client and attacker. We move the attacker from far away (above three meters) to a medium distance (two meters and 50 cm) to very close (10 cm and five centimeters) to see how SecureArray performs. The results presented here are averaged across different locations and different time gaps. As shown in Figure 16, we observe that when the client and attacker get closer, the client false alarm rate is more or less the same while the attack miss rate slightly increases. This is expected as the peaks on the AoA signatures reflecting the direct path signal (if not blocked) become similar when the client and attacker are very close to each other. This brings in some degree of similarity between the two AoAs and increases the similarity metric especially when the direct path signal is stronger than all the reflection path signals. We can see that the attack miss rate is zero for large and medium distances and increases slightly to 3.7%, which is still well within an acceptable level when the client and attacker are only five centimeters to each other. Please note that it's unlikely that an attacker can be placed at exactly the same location (0 cm distance) as the client when the client is static.

4.2.5 *Inter-packet time*

The channel is dynamic between any particular antenna pair in both amplitude and phase. In terms of our AoA signature stability, the phases of most antennas at the array need to change at the same time to the same direction in order to show a obvious effect on the lobe peak positions on an AoA signature. From Figure 17, we can

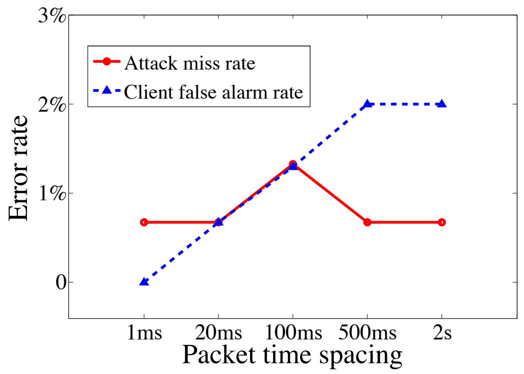

Figure 17: Performance of SecureArray with different time gaps between the packets obtained for signature comparison from client-client and client-attacker, for static senders.

see that in a typical working office environment when the clients are static, with the inter-packet time gap increased from one millisecond to two seconds, SecureArray still performs well, with the client false alarm rate increased slightly to around 2%, due to human movements in the environment, we believe. Note that the attack detection rate is always high because the attacker's signature does not tend to change to be similar to the legitimate client's signature with time. So the attack detection rate is not affected by the large time gaps between packets employed for comparison. However, movements in the environment do change the clients' signatures, which is reflected in the slightly increased false alarm rate.

4.3 Mobile experiments

In this section, we evaluate SecureArray's performance for mobile clients. The two Soekris boxes serving as the attacker and legitimate client are held by the experimenter with a separation of around 60 cm (this value varies during the walking process with natural human movements) away from each other. We test SecureArray with the experimenter walking along the three routes labeled as dashed lines in Figure 10. To measure the attack detection rate, signatures generated from packets recorded at the attacker and client with different time gaps are compared. All the packets are recorded when the experimenter is walking along the trajectories. To measure the false alarm rate, two signatures generated from packets both recorded at the client with different time gaps are employed for comparison. We choose one route to be in the same office with a strong line-of-sight to the AP and the other two routes with many non-line-of-sight portions. The experimenter walks at a speed of around 4 kph so the expected coherence time is around 12 ms.

We compare the attack detection rate and client false alarm rate with respect to different time gaps between the packets employed for signature comparison. The results are shown in Figure 18, varying the time gaps from 200 ms to one millisecond. We can see that the attack detection rate is always high, roughly 100%. However, when the time gap is 200 ms and 50 ms, the client false alarm rate is 100% and 96.7% which means the legitimate client itself is hardly ever identified with such big time gaps when client is moving. When the time gap decreases to around 20 ms, we have the false alarm rate decreased to 46.7% but still high. When the time gap becomes 5 ms below the wireless coherence time, the false alarm rate decreases dramatically to 6.7% and this value further decreases to 0% for one millisecond time gap. One point we note here is that, the phase-calibrated version is a special candidate among all the perturbation groups. The false alarm rate is lower when applying

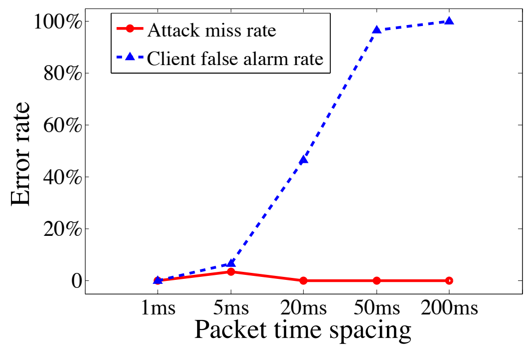

Figure 18: Performance of SecureArray with different time gaps for mobile clients in our testbed.

the correct calibration phases especially in line-of-sight scenarios, as the direct path is strong and relatively stable compared with reflection paths during movements, which increases our metric value and mitigates the false alarm error rate. When the AP has no idea of the mobility status of the client, the safe option is to include this perturbation group in for averaging. Nowadays, a lot of devices have a variety of different sensors built in and when a device starts moving, it can be easily detected at the client side and at the AP side. If the AP has the information of client's mobility status, it can choose intelligent random perturbation options. We will explore possible schemes to reduce the false alarm rate for mobile clients when the packets employed for comparison are not recorded within the coherence time in our future work. We believe one promising direction is to explore the stability nature of the direct path peak on the AoA signature when the client is moving.

4.4 System latency

System latency is an important characteristic for any real-time system. As SecureArray works with any kind of packet and any part of the packet, we choose to use the most robust preamble part for our processing. The preamble itself contains hundreds of samples when sampled at 40 MHz rate. In principle, one sample would suffice, but because of noise, it's preferable to employ multiple samples and average the correlation matrix to achieve a more stable AoA signature. As shown in prior work [39], 10 samples suffice for AoA signature stability. Once SecureArray detects the preamble and records this number of samples, it begins processing while the rest of the packet is still on the air. The components of system latency (measured from the start of the second packet in a comparison) are described below:

- T_1: time taken for packet detection and samples recording with WARP. If one short training symbol in the preamble is employed for packet detection, T_1 is 1.6 μs.
- T_2: time taken for samples to be transferred to the server. As we employ 10 samples for processing, we have 80 samples to be transferred with a total of eight radio boards at the transfer rate of 1 Mbps supported by the WARP second generation platform: $T_2 = \frac{(80\text{ samples})(32\text{ bits/sample})}{1\text{ Mbit/s}} = 2.56$ ms.
- T_3: time taken for the server to compute the metric and make the decision. For an eight-antenna array, our AoA signature generation involves eigenvalue decomposition and processing of 8x8 matrices. The peak finding and comparison algorithm run very fast. More computation is needed for a larger L value. However, these are independent processes and can be run in parallel. For our current Matlab implementation with an Intel Core i7 2.30 GHz CPU and 8 GB of RAM, the processing time is in the range of 10 ms to 20 ms with $L = 5$.

Therefore, the total latency that our prototype incurs starting from the start of the second packet (excluding bus latency) is around 20 ms. We expect a full FPGA implementation to operate orders of magnitude faster.

5. RELATED WORK

Xiong *et al.* have proposed the use of AoA signatures as a fingerprinting mechanism [38] to be used in wireless networks. SecureArray expands upon this work by proposing new signal processing techniques for enhancing AoA signature specificity, integrating AoA signature checking into the existing 802.11 security protocol framework, introducing the DataCheck (§2.2.2) protocol for mitigating authenticated spoofing in the event of user password compromise, and providing a comprehensive security evaluation measuring false alarm and miss rates (§4).

SecureArray is designed to operate alongside traditional protocol-based security, which has a history of first becoming widely-deployed and then being compromised. In the light of WEP's well-known security flaws (§1), the 802.11 specification was amended in 2004 (802.11i) to include enhanced security [1]. In 2009, the 802.11 specification was again amended (802.11w) to include protection for deauthentication, disassociation, and certain other management frames [2]. The 2012 revision of IEEE 802.11 [3] incorporates this amendment. Nonetheless, several attacks on this revision exist [9, 14, 4] and are mitigated by SecureArray as described above (§2). We again note that SecureArray only mitigates active attacks. Other MIMO-based schemes such as STROBE [5] have recently been proposed to mitigate passive eavesdropping; these schemes complement our work.

Localization systems. In general, an accurate localization system [39, 40] could serve as a proxy for SecureArray fingerprints, but a key advantage of SecureArray over most RF-based location systems is that it only requires the involvement of one AP. PinLoc [32] is a recent localization system that uses per-OFDM subcarrier channel measurements (CSI). This is made possible on a commodity NIC by a CSI measurement tool previously released by Halperin [21]. PinLoc uses the phase information in CSI measurements along with amplitude to fingerprint clients and then localize them in a one meter by one meter grid. One could imagine PinLoc being co-opted for security purposes, but this would in fact be undesirable from a security standpoint, because an attacks may originate centimeters away from the victim (consider, for example, an attacker surreptitiously placing a mobile phone in the victim's handbag). Like many other systems, PinLoc requires a a time-intensive environmental calibration step to learn location signatures. SecureArray requires no environmental calibration.

PHY-layer channel fingerprinting. The concept of fingerprinting clients based on physical layer channel information has been explored in many ways in the wireless systems literature. In one of the first such designs, Faria and Cheriton propose the use of a "signalprint" [16], the vector of received signal strengths (RSS) that a predetermined set of nearby APs observe from a wireless client's transmission. In later work, Chandrasekaran *et al.* use RSS to detect spoofing, achieving 10% false alarm and missed attack rates when the two are equalized [12]. AoA information is much more detailed than RSS, and so SecureArray can achieve orders of magnitude better specificity and selectivity over RSS-based systems with just one AP, as is common in home settings, obviating the need for multiple nearby APs or any coordination of differently-administered APs.

CSITE [24] is a recent system that is perhaps closest in spirit to SecureArray; it uses CSI magnitude measurements averaged in time over multiple frames to form a client signature. There are two main differences between SecureArray and CSITE. Firstly, SecureArray uses AoA information at the AP instead of averaged CSI magnitudes. As Patwari and Kasera observe in the context of RSS, however, approaches based on channel strength can be subverted by attackers with directional or phased-array antennas [29]. Since RSS is simply CSI magnitude information averaged across different subcarriers, per-subcarrier CSI approaches such as CSITE can be subverted by attackers with phased array software-defined radios capable of transmitting with independent beamforming weights on different subcarriers. Secondly, CSITE uses multi-packet time-averages of channel magnitude information, which yields a less selective discriminator with missed intrusion attempts of up to 5% and false alarm rates of up to 60%, depending on the traffic intensity of the legitimate client.

Other candidates for wireless channel signatures at the physical layer include time of flight [28], signal-to-noise ratio measured at an AP [15], client-AP channel impulse response [29], and combination of multi-tonal probes and channel impulse response [41]. SecureArray has an intrinsic advantage over these approaches because it is extremely difficult for an attacker to generate a signal that arrives at the AP from the same physical directions as the legitimate client without being co-located with the client to within small fractions of the RF wavelength (*i.e.*, 12 cm at 2.4 GHz).

PHY-layer client fingerprinting. The problem of fingerprinting 802.11 clients themselves (in contrast to fingerprinting the channel between client and AP as do SecureArray and other prior work below) has been thoroughly explored in the literature. PARADIS [11] is a prime example of this type of device fingerprinting system that achieves 99% accuracy. Though this device fingerprinting can improve security, the solution is qualitatively different because an attacker can simply acquire a legitimate client's 802.11 device to launch her attack. Other candidates for device fingerprinting include radio transmission startup transient signal envelope [7], the 802.11 probe request interarrival time distribution [26], and client CPU clock skew [25, 23].

Link-layer client fingerprinting. These techniques encompass examining MAC (hardware) addresses, active probe requests for cached SSIDs, and requests for a certain set of network servers from clients [20, 27]. SecureArray has an advantage over such link-layer techniques in that it is reliable even if clients take countermeasures, such as reprogramming their hardware addresses, cached SSIDs, and encrypting their network-layer traffic.

6. DISCUSSION

1. SecureArray overhead: SecureArray introduces very limited overhead because the operation is triggered only when suspicious packets are received or unusual behaviors are detected. SecureArray is not trying to update the AoA information continuously at the frequency decided by the coherence time.

2. Forge the legitimate client's AoA signature: It's very difficult for the attacker to forge the legitimate client's AoA signature even if it's equipped with an advanced phased array. There are multiple schemes the attacker may employ:

- The attacker is placed at exactly the same location as the legitimate client. The attacker may have a chance to be located near to the client. However, it is unlikely for the attacker to be placed at the same physical location and furthermore, the attacker usually has a preference to be located far away in order not to be discovered. A small five centimeters location difference causes a significant AoA signature change which can be easily detected.

- The attacker obtains the legitimate client's AoA signature and then employ an antenna array to mimic the client's signature while the attacker is at a different location. We argue that this is difficult to be realized. In order for the attacker to obtain the client's AoA signature at the AP, the attacker needs to have the same number of antennas and all the antenna to be at the same positions as the AP. Even we assume the AP has successfully obtained the client's AoA signature, it's still challenging for the attacker to mimic this signature at a different location. If the attacker is equipped with an antenna array, the attacker can utilize it to form different radiation patterns. However, the radiation pattern at the attacker side is not the AoA signature generated at the AP. The AoA signature at the AP is decided by both the radiation pattern and the physical prorogation paths between the attacker and AP. It's almost impossible for the attacker to create a same physical environment at a different location. The dynamic nature of the environment makes it even more challenging.

- When the client is moving, the attacker tries to predict the client's future location. We argue that it's challenging to predict a precise future location of a mobile client. A rough location estimate does not fail SecureArray as we show experimentally in Figure 16 that even a five centimeter small location difference presents a very high detection rate.

7. CONCLUSION

We have described SecureArray, a system that leverages multipath propagation in the wireless channel to construct highly specific client signatures, just as MIMO leverages multipath to increase capacity. Our testbed results show that in a typical office environment, SecureArray is able to mitigate 100% of WiFi spoofing attack attempts while at the same time triggering false alarms on just 0.6% of legitimate traffic. Detection rate remains high even when the attacker is located very close to the legitimate client. The performance of SecureArray degrades very little with fewer numbers of antennas on the AP and when clients are mobile.

To identify wireless protocol exploits, the security community uses model checking tools such as SPIN [22], augmented with timing information. We anticipate that the tests we describe above (§2.2) will in the future be integrated into such model checkers so that SecureArray can check all testable points (places where a legitimate client transmits two frames within a coherence time) in the protocol for attack activity.

For mobile clients, the direct path is relatively stable compared with the reflection paths. By identifying the stable direct path peak on the AoA signatures when the client is moving, heavier weight can be assigned to the direct path bearing for signature comparison. We believe with smart process, SecureArray may be able to break the coherence time constraint and work with packets recorded within a larger time interval so the application scope of SecureArray can be extended.

8. ACKNOWLEDGEMENT

This material is based on work supported by the European Research Council under Grant No. 279976. Jie Xiong is supported by the Google European Doctoral Fellowship in Wireless Networking. We would like to thank Prof. Sneha Kumar Kasera, as well as the anonymous reviewers, for their valuable feedback on this paper.

9. REFERENCES

[1] IEEE Standard 802.11i-2004: MAC Security Enhancements, June 2004.

[2] IEEE Standard 802.11w-2009: Protected Management Frames, Sept. 2009.

[3] IEEE Standard 802.11: Wireless LAN MAC and PHY Layer Specifications, Mar. 2012.

[4] M. Ahmad and S. Tadakamadla. Short paper: security evaluation of IEEE 802.11w specification. In *Proceedings of ACM WiSec*, 2011.

[5] N. Anand, S. Lee, and E. Knightly. STROBE: Actively securing wireless communications using zero-forcing beamforming. In *Proc. of IEEE Infocom*, 2012.

[6] E. Aryafar, N. Anand, T. Salonidis, and E. Knightly. Design and experimental evaluation of multi-user beamforming in wireless LANs. In *Proc. of ACM MobiCom*, 2010.

[7] M. Barbeau, J. Hall, and E. Kranakis. Detecting impersonation attacks in future wireless and mobile networks. *Springer Lecture Notes in Computer Science*, 4074(2006):80–95, Aug. 2006.

[8] M. Beck and E. Tews. Practical attacks against WEP and WPA. In *Proc. of ACM WiSec*, pages 79–86, 2009.

[9] B. Bertka. 802.11w security: DoS attacks and vulnerability controls. In *Proc. of Infocom*, 2012.

[10] A. Bittau, M. Handley, and J. Lackey. The final nail in WEP's coffin. In *Proc. of IEEE Symp. on Security and Privacy*, 2006.

[11] V. Brik, S. Banerjee, M. Gruteser, and S. Oh. Wireless device identification with radiometric signatures. In *Proc. of ACM MobiCom*, 2008.

[12] G. Chandrasekaran, J. Deymious, V. Ganapathy, M. Gruteser, and W. Trappe. Detecting identity spoofs in 802.11e wireless networks. In *Proc. of IEEE Globecom*, 2009.

[13] M. Eian and S. Mjølsnes. The modeling and comparison of wireless network denial of service attacks. In *Proc. of ACM MobiHeld*, 2011.

[14] M. Eian and S. Mjølsnes. A formal analysis of IEEE 802.11w deadlock vulnerabilities. In *Proc. of IEEE Infocom*, 2012.

[15] D. Faria and D. Cheriton. No long-term secrets: Location based security in overprovisioned wireless LANs. In *Proc. of ACM HotNets*, 2004.

[16] D. Faria and D. Cheriton. Radio-layer security: Detecting identity-based attacks in wireless networks using signalprints. In *Proc. of ACM WiSe*, 2006.

[17] S. Gollakota and D. Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. In *Proc. of ACM SIGCOMM*, 2008.

[18] S. Gollakota, S. Perli, and D. Katabi. Interference alignment and cancellation. In *Proc. of ACM SIGCOMM*, Aug. 2009.

[19] D. Goodin. Lax security led to TJX breach. *The Register*, May 2007. http://www.the-register.co.uk/2007/05/04/txj_nonfeasance.

[20] M. Gruteser and D. Grunwald. Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitative analysis. In *Proc. of ACM WMASH*, 2003.

[21] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. Tool release: Gathering 802.11n traces with channel state information. *SIGCOMM CCR*, 41(1):53, Jan. 2011.

[22] G. Holzmann. The model checker SPIN. *IEEE Trans. Software Eng.*, 23(5):279–295, 1997.

[23] S. Jana and S. K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. In *Proceedings of ACM MobiCom*, 2008.

[24] Z. Jiang, J. Zhao, X. Li, J. Han, and W. Xi. Rejecting the Attack: Source Authentication for Wi-Fi Management Frames using CSI Information. In *Proc. of IEEE Infocom*, 2013.

[25] T. Kohno, A. Broido, and K. Claffy. Remote physical device fingerprinting. In *Proc. of IEEE Symp. on Sec. and Priv.*, pages 211–225, May 2005.

[26] D. C. Loh, C. Y. Cho, C. P. Tan, and R. S. Lee. Identifying unique devices through wireless fingerprinting. In *Proc. of the ACM WiSec Conf.*, pages 46–55, Mar. 2008.

[27] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall. 802.11 user fingerprinting. In *Proc. of ACM MobiCom*, 2007.

[28] P. Papadimitratos, M. Poturalski, P. Schaller, P. Lafourcade, D. Basin, S. Čapkun, and J. Hubaux. Secure neighborhood discovery: a fundamental element for mobile ad hoc networking. *IEEE Comm. M.*, 46(2):132–139, Feb. 2008.

[29] N. Patwari and S. Kasera. Robust location distinction using temporal link signatures. In *Proc. of the ACM MobiCom Conf.*, pages 111–122, Sept. 2007.

[30] R. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Trans. on Antennas and Propagation*, AP-34(3):276–280, Mar. 1986.

[31] B. Schneier, Mudge, and D. Wagner. Cryptanalysis of Microsoft's PPTP authentication mechanisms (MS-CHAPv2), Oct. 1999.

[32] S. Sen, B. Radunovic, R. Choudhury, and T. Minka. Spot localization using PHY layer information. In *Proceedings of ACM MobiSys*, 2012.

[33] Y. Sheng, G. Chen, D. Kotz, and A. Campbell. Detecting 802.11 MAC layer spoofing using received signal strength. In *Proc. of IEEE Infocom*, pages 1768–1776, Apr. 2008.

[34] C. Shepard, H. Yu, N. Anand, L. Li, T. Marzetta, R. Yang, and L. Zhong. Argos: Practical many-antenna base stations. In *Proc. of ACM MobiCom*, 2012.

[35] R. Steele. *Mobile Radio Communications*. IEEE Press, 1994.

[36] K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, and G. Voelker. SAM: Enabling practical spatial multiple access in wireless LAN. In *Proc. of ACM MobiCom*, 2009.

[37] E. Tews, R.-P. Weinmann, and A. Pyshkin. Breaking 104-bit WEP in less than 60 seconds. *Springer Lecture Notes in Computer Science*, 4867:188–202, Jan. 2008.

[38] J. Xiong and K. Jamieson. SecureAngle: Improving wireless security using angle-of-arrival signatures. In *Proc. of ACM HotNets*, 2010.

[39] J. Xiong and K. Jamieson. ArrayTrack: A Fine-Grained Indoor Location System. In *Proc. of NSDI*, 2013.

[40] M. Youssef and A. Agrawala. The Horus WLAN location determination system. In *Proc. of ACM MobiSys*, 2005.

[41] J. Zhang, M. H. Firooz, N. Patwari, , and S. K. Kasera. Advancing wireless link signatures for location distinction. In *Proceedings of ACM MobiCom*, 2008.