Acoustic Integrity Codes: Secure Device Pairing Using Short-Range Acoustic Communication

Florentin Putz
Secure Mobile Networking Lab
Department of Computer Science
TU Darmstadt, Germany
fputz@seemoo.de

Flor Álvarez
Secure Mobile Networking Lab
Department of Computer Science
TU Darmstadt, Germany
falvarez@seemoo.de

Jiska Classen
Secure Mobile Networking Lab
Department of Computer Science
TU Darmstadt, Germany
jclassen@seemoo.de

ABSTRACT

Secure Device Pairing (SDP) relies on an out-of-band channel to authenticate devices. This requires a common hardware interface, which limits the use of existing SDP systems. We propose to use short-range acoustic communication for the initial pairing. Audio hardware is commonly available on existing off-the-shelf devices and can be accessed from user space without requiring firmware or hardware modifications.

We improve upon previous approaches by designing *Acoustic Integrity Codes (AICs)*: a modulation scheme that provides message authentication on the acoustic physical layer. We analyze their security and demonstrate that we can defend against signal cancellation attacks by designing signals with low autocorrelation. Our system can detect overshadowing attacks using a ternary decision function with a threshold. In our evaluation of this SDP scheme's security and robustness, we achieve a bit error ratio below 0.1% for a net bit rate of 100 bps with a signal-to-noise ratio (SNR) of 14 dB. Using our open-source proof-of-concept implementation on Android smartphones, we demonstrate pairing between different smartphone models.

CCS CONCEPTS

• **Security and privacy** → **Authentication**; **Mobile and wireless security**; Security protocols; Key management; • **Networks** → Mobile networks; Wireless access networks; Cyber-physical networks; • **Hardware** → Digital signal processing.

KEYWORDS

physical-layer security, signal cancellation, secure device pairing, acoustic communication, trust, integrity codes, android

ACM Reference Format:
Florentin Putz, Flor Álvarez, and Jiska Classen. 2020. Acoustic Integrity Codes: Secure Device Pairing Using Short-Range Acoustic Communication. In *13th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '20), July 8–10, 2020, Linz (Virtual Event), Austria.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3395351.3399420

WiSec '20, July 8–10, 2020, Linz (Virtual Event), Austria

ACM ISBN 978-1-4503-8006-5/20/07…$15.00
https://doi.org/10.1145/3395351.3399420

1 INTRODUCTION

An increasing number of ubiquitous computing devices require secure provisioning and pairing mechanisms. During the setup of cyber-physical systems, consumers still struggle with constructing secure communication channels, as those require preexisting *security contexts*, e.g. shared public keys when using asymmetric cryptography. Establishing such a prior security context is a critical step to ensure the communication's security.

One way of establishing a new security context is SDP. The devices pair in an ad-hoc manner and establish an authenticated key. In contrast to public key infrastructures (e.g., x.509 or OpenPGP), SDP does not require any *trusted third parties*, where Alice has to trust other entities that help her initialize a security context with Bob. Therefore, SDP is well-suited for offline and private scenarios, emergency settings, or bootstrapping new deployments that are not part of any public key infrastructure. There is growing research in using *physical device proximity* to support SDP, by either using a location-limited communication channel or extracting keys from measuring the environment [17].

SDP requires a common hardware interface that both devices use for pairing, which limits the applicability of existing SDP schemes. Many SDP schemes use displays, cameras, vibration motors, accelerometers, infrared transducers, or wireless near-field communication [7, 8, 17]. A commonly available hardware interface that is rarely used for SDP in practice is audio via speakers and microphones. *Acoustic communication* requires only minimal user interaction, which increases usability and reduces potential failure points. In contrast to electromagnetic wireless communication such as Wi-Fi or Bluetooth, acoustic communication requires no complex network configuration and can be implemented as user space software, even with physical-layer capabilities. This allows us to perform acoustic communication on existing off-the-shelf devices without hardware modification, reducing deployment costs [21].

Previous approaches that used audio for SDP, such as "HAPADEP" [40], require a manual verification phase for security reasons, which is error-prone and reduces usability. We design a secure acoustic communication protocol that requires less security-critical user interaction by incorporating the recent research direction of *physical-layer security*. These techniques consider security at the lowest layer using the physical properties of the wireless radio channel. Specifically, we use *Integrity Codes*, which were proposed by Čapkun et al. to provide message integrity in the presence of active attackers on the radio channel [3]. Using Integrity Codes, we eliminate the need for a separate verification step, which speeds up the pairing process and increases usability. Integrity Codes can be vulnerable to *signal cancellation* attacks [9, 16, 27]. We therefore

analyze this threat and propose countermeasures. Our resulting design improves Integrity Codes by mitigating signal cancellation attacks.

Our main contribution is the design, implementation and evaluation of Acoustic Integrity Codes (AICs), which we use for SDP. This work combines the independent research fields of SDP, acoustic communication, and physical-layer security. To the best of our knowledge, Integrity Codes have not been applied to acoustic communication before. Our individual contributions are:

- **Analysis of Signal Cancellation Attacks.** We show that signal cancellation attacks fail for signals with low autocorrelation. We propose system parameters that improve Integrity Codes.
- **Design of Acoustic Integrity Codes.** We use Integrity Codes to secure acoustic communication.
- **Evaluation.** We evaluate the security and robustness of AICs using simulations.
- **Design of an SDP scheme using AICs.** We apply AICs to design an acoustic SDP scheme.
- **Implementation of a prototype for modern Android devices.** We implement an open-source proof-of-concept for Android smartphones.

Our work is structured as follows: After introducing related work and Integrity Codes in section 2, we present our design in section 3 and our implementation in section 4. Then, we analyze the security of AICs in section 5 and evaluate them in section 6. Finally, we conclude our work in section 7.

2 RELATED WORK

In this section, we describe SDP and present related work using acoustic communication to perform SDP. We also introduce Integrity Codes.

2.1 Secure Device Pairing

SDP enables multiple devices with no prior security context to establish a secure communication channel over an untrusted channel. We only consider two devices A and B belonging to Alice and Bob, respectively. The devices want to communicate over an a priori insecure communication channel. They use an *out-of-band (OOB) channel* to authenticate a key exchange, which they can then use to construct a secure communication channel using a standard cryptographic protocol such as Transport Layer Security (TLS).

The audio channel can be used as a *location-limited channel* to perform this key exchange [1]. Goodrich et al. [11, 12] developed an SDP system called "Loud&Clear", which requires the user to detect whether two computer-generated speech sequences are identical. Soriente et al. [40] presented "HAPADEP", in which the devices encode their public keys as short audible melodies that the other device can decode. They use a second verification phase to detect machine-in-the-middle (MITM) attacks, which requires active user participation. Halperin et al. [14] developed an SDP system called "Zero-Power Sensible Key Exchange" involving acoustic communication for implantable medical devices (IMDs). The IMD generates a symmetric session key and transmits it as an audible sound wave to the external device via a piezo element. This OOB channel has to be secret, since their system lacks eavesdropping protection. Halevi and Saxena [13] showed that eavesdropping is possible even with off-the-shelf equipment, using digital signal processing. Claycomb and Shin [4] devised an acoustic SDP method called "UbiSound", which uses a single unidirectional audio transmission. The user is responsible for aborting the pairing process in case of malicious interference. Mayrhofer et al. [22] presented "UACAP" as a general SDP implementation that is designed to support multiple OOB channels such as 2D barcodes, manual string comparison, and audio (based on "HAPADEP"). Han et al. [15] proposed the SDP protocol "MVSec", using either an audio or a visual channel as the OOB channel to pair smartphones with cars. This pairing protocol happens mainly over an in-band Bluetooth channel. The audio channel is used as the OOB channel to bidirectionally transfer truncated commitments to the public keys.

In contrast to our design, these SDP schemes usually require additional user interaction to defend against active attackers. To the best of our knowledge, there is no publicly available acoustic SDP implementation for current iOS or Android devices.

Apart from the audible sound spectrum, the inaudible ultrasound spectrum has also been used as part of the secure pairing process [19, 23, 24]. The ultrasound spectrum, however, is not suitable to secure commercial off-the-shelf devices, because this requires additional hardware. Apart from using acoustic communication directly, previous research on SDP also utilized the audio channel for demonstrative identification [28], as part of an audiovisual pairing scheme [30], or for ambient sensing [25, 33, 36].

2.2 Integrity Codes

Over the last decade, the research community has investigated whether security goals such as *authentication* and *integrity* protection can be realized on the physical layer. As these physical-layer security techniques usually do not assume a prior security context, they are well-suited to protect SDP. In this section, we introduce the *Integrity Code* physical-layer security primitive [3], which we use to secure acoustic communication.

Čapkun et al. established the concept of *Integrity Codes ("I-Codes")* [3], which is a modulation scheme that protects the message integrity on the wireless physical layer without requiring any shared key material. Instead, integrity and authentication can be protected when the receiver knows that:

(1) the sender is currently transmitting and
(2) the sender is in the receiver's range.

Figure 1 illustrates Integrity Codes. The transmitter first applies a unidirectional error code (e.g., Manchester Code or Berger Code), which can detect one direction of bit flips (from $0 \rightarrow 1$). Then, the

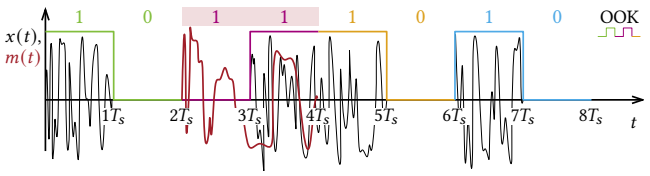


Figure 1: Integrity Code signal representing the data 1011. An attacker interferes using the red signal. This flips the third bit from $0 \rightarrow 1$, which can be detected.

transmitter performs on-off keying (OOK) using the encoded data. OOK is a form of amplitude-shift keying (ASK), where a 0 is represented by the absence of a carrier wave and a 1 is represented by the presence of a carrier wave. Instead of using a deterministic carrier signal, however, we use a stochastic signal in each on slot. The idea is that random signals cannot be cancelled by an active attacker via destructive interference. An attacker is not able to change any bit from $1 \longrightarrow 0$. Any other modification of the message can be detected at the receiver using unidirectional error codes. Integrity Codes have been used to assist in SDP with radio communication [10, 37]. We use Integrity Codes to secure acoustic communication.

3 DESIGN

In this section, we present the design of our SDP scheme, which uses short-range acoustic communication. We design AICs to secure this communication on the physical layer. Our main goal is to securely transmit public key material d (or shorter commitments such as hash values) from Alice's device A to Bob's device B, even in the presence of an active adversary Mallory, who tries to manipulate this communication using her devices M_n. We focus on unidirectional SDP for the private and social application classes and consider pairing in the other direction as an optional subsequent but separate step that works in the same way. The public key material can then be used to initialize a security context between the devices.

Specifically, our system shall achieve *message authentication* of the transmitted public key material d, which consists of the following two security properties [18, p. 25]:

- **Identification of the sender:** B is able to verify whether the message d originated from A.
- **Integrity:** B can detect whether the message d was modified during transmission. This is implied by the first property, since then A would no longer be the message's originator.

Confidentiality or availability protection is out of scope.

We use an *acoustic channel* as the physical channel, i.e., we transmit information by modulating a mechanical pressure wave generated by A using a speaker [7]. The receiver B records this

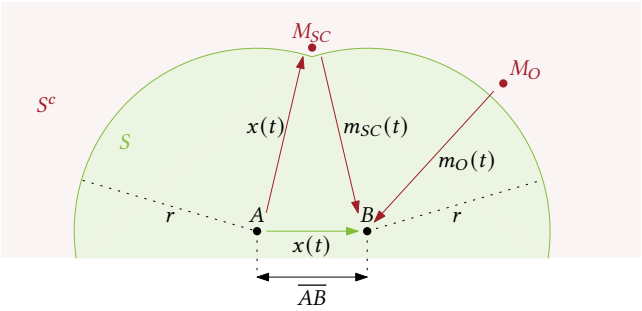


Figure 2: This system model shows a top-down perspective of the devices A and B with the safe area S around them (green). Possible adversaries can perform signal cancellation attacks with device M_{SC} (see subsection 5.1) or overshadowing attacks with device M_O (see subsection 5.3) in the insecure area S^c (red) outside the secure area S.

using a microphone. Figure 2 shows the devices A and B and their environment during the pairing process. We require that Alice and Bob perform the pairing while being in proximity, such as by standing next to each other. We denote the distance between the devices A and B as $\overline{AB}$. We denote the immediate area around Alice and Bob as S, the *safe area*. The remaining *unsafe area* is S^c. We model S using two spheres centered at A and B, respectively, with radius r.

3.1 Assumptions

We assume that Alice and Bob control the safe area, i.e., there are no malicious devices in S. We argue that this assumption is realistic for the private application class such as when pairing devices at home. For the social application class, this means that pairing should not be performed in crowded areas, where an attacker could be close. For the model parameters, we assume $r > 40\,\text{cm} > \overline{AB}$. These parameters can be adjusted depending on the specific use case. We study the security implications in section 5. We assume that device A is equipped with a speaker and that device B is equipped with a microphone. Microphones and speakers are commonly available on commercial off-the-shelf devices. We assume that the devices A and B are not compromised, i.e., the software and hardware performing the SDP scheme are not controlled by Mallory. Finally, we assume that A and B know the public protocol parameters, which we explain in the next sections.

3.2 Adversary Model

Making realistic assumptions on the attacker's capabilities is crucial to effectively design a secure system [6]. Whereas weak attacker models can underestimate the threats, a very strong attacker model can lead to an overly complicated system design or require more advanced hardware, which could hinder adoption [29]. We make practical and realistic assumptions.

Research on wireless network security often uses a *Dolev-Yao attacker model* [5], which assumes a very strong attacker, who is able to control and manipulate all messages on the network. It is, however, not necessarily realistic to assume that the attacker can *freely* modify or annihilate the wireless signals at the receiver's antenna [29]. We therefore use a weaker but more realistic Dolev-Yao attacker model, which is typically used when applying Integrity Codes [3, 10, 37].

The main goal of the attacker (Mallory) is to impersonate Alice. Mallory wants that Bob accepts her key instead of Alice's key. Mallory can *eavesdrop* all signals (passive). She can also *send her own signals* (active), which superimpose with the legitimate signal at the receiver's antenna. Mallory's signal transmissions are still bound by the same physical signal propagation laws that also govern legitimate transmissions, i.e., they arrive at the receiver after a propagation delay with a phase shift. We assume that Mallory can only operate outside the secure area S, which is controlled and observed by Alice and Bob via proximity. We assume that she cannot trivially disable the communication channel by shielding A's signals from propagating to B with a physical barrier.

Mallory may control any number N of devices $\{M_n : n \in \{1, \ldots, N\}\}$, which are placed anywhere outside the secure area S. We denote as $x(t)$ the signal that A transmits to B. We denote as

$m_n(t)$ the signal that M_n transmits. These signals are affected by the acoustic channel H, which attenuates and delays the signal. We also consider additional noise $v(t)$. Then, B receives the following superposition of all these signals:

$$y(t) = \underbrace{H_{A\to B}\{x(t)\}}_{\text{legitimate signal } x'(t)} + \sum_{n=1}^{N} \underbrace{H_{M_n\to B}\{m_n(t)\}}_{\text{attacker } m'_n(t)} + \underbrace{v(t)}_{\text{noise}} \quad (1)$$

When designing our communication system, we account for signal cancellation, bit flipping, and overshadowing attacks. We analyze these types of attacks in section 5.

3.3 Secure Device Pairing Scheme

Alice uses her device A to transmit some public key material d to Bob's device B. The pairing process consists of the following steps:

(1) Alice initializes the pairing process on her device A. The device now repeatedly broadcasts d on the acoustic channel using AICs.
(2) Alice tells Bob that he can start receiving data now.
(3) Bob accepts the pairing process on his device B.
(4) Device B receives the key material over the acoustic channel.
(5) Device B notifies Bob that it received the key. The transmission was either successful or there was an error due to background noise or an attacker.
(6) Bob tells Alice that he finished the pairing process.
(7) Alice stops the transmission on her device A.

The SDP process is successful if there was no transmission error. If the environmental noise is too high or if there is an attacker, the transmission fails and they can try again at another location. We design AICs to provide message authentication of the communication on the physical layer.

AICs require that the receiver is aware of an ongoing transmission. We could use additional signaling on the physical layer to automate the manual steps (2), (3), (6) and (7), but this signaling could be modified by the attacker. We cannot secure this signaling, since we assume that we have no prior security context. We also cannot use AICs to secure this signaling, since AICs require that the receiver always knows that the legitimate transmitter is active.

3.4 Acoustic Integrity Codes

We now explain how we secure acoustic communication using the AIC modulation scheme. This is the foundation of our SDP scheme. AICs apply the concept of Integrity Codes [3] to acoustic signals. We use Integrity Codes to defend against signal cancellation and overshadowing attacks.

Figure 3 illustrates transmission and reception of AICs. The transmitter A encodes the data using unidirectional error coding (e.g., Manchester Coding) and frames the data by prepending a delimiter header $D[n] = (1, 1, 1, 0, 0, 0)$. Čapkun et al. have shown that this delimiter is optimal [3]. After framing, A converts the time discrete sequence $b[n]$ (consisting of repeated frames) into a time continuous baseband signal $s(t)$ using baseband OOK. As a result, each bit $b[n]$ corresponds to a time slot of duration T_s during which $s(t)$ has a constant value of either 0 or 1. The signal's gross bit rate, including coding overhead and the delimiter, is $R_g = 1/T_s$. The net bit rate $R_n \approx R_g/2$ describes the effective number of bits that can be transmitted per second.

Finally, A generates the bandpass signal $x(t)$ in the frequency band $[f_{\text{low}}, f_{\text{high}}]$. Instead of using a deterministic carrier signal, A modulates a *stochastic "carrier" signal* $w(t)$, sampled from a random process $\{W(t)\}$. Unless otherwise noted, we use a white Gaussian noise process $\{W_{\text{WGN}}(t)\}$. This step differs from conventional modulation schemes such as ASK, where $s(t)$ is used to modulate a deterministic carrier signal. This randomness is essential for integrity protection, as stochastic signals with low autocorrelation cannot be cancelled out by an attacker (see section 5).

After modulation, A transmits the signal $x(t)$ through the acoustic channel $H_{A\to B}$ using a speaker:

$$x'(t) = H_{A\to B}\{x(t)\} = \alpha_A x(t - \tau_A) \quad (2)$$

In our system model, we expect to have a strong line-of-sight (LOS) component due to the devices' proximity. Our approximate channel model accounts for the attenuation α_A and propagation delay τ_A on the LOS path. The propagation delay is proportional to the distance between the devices and satisfies the relation $\tau_A = \overline{AB}/c_s$, where c_s is the speed of sound in our transmission medium. The transmission is also subject to additive noise $v(t)$ resulting from sound sources in the environment and from thermal noise in the electrical components.

B records and receives the resulting signal $y(t)$, as shown in Equation 1. After filtering out background noise using a bandpass filter, B performs synchronization to recover the frame boundaries based on their delimiter and then demodulates this signal using a decision function. In conventional modulation schemes such as ASK, we would not consider an attacker at the physical layer and therefore always decide on one of two possible states $S_{\text{binary}} = \{0, 1\}$, e.g., based on maximum likelihood. Such a decision function D_{binary}, however, is vulnerable to an overshadowing attack, which we analyze in subsection 5.3. Instead, our decision function D_{ternary} considers an attacker at the physical layer and decides on one of three possible states $S_{\text{ternary}} = \{0, 1, \varepsilon\}$, where ε signals an error:

$$D_{\text{ternary}}(p_1, p_2) = \begin{cases} 0, & \text{if } p_1 < P_{\text{th}} \text{ and } p_2 > P_{\text{th}} \\ 1, & \text{if } p_1 > P_{\text{th}} \text{ and } p_2 < P_{\text{th}} \\ \epsilon, & \text{otherwise.} \end{cases} \quad (3)$$

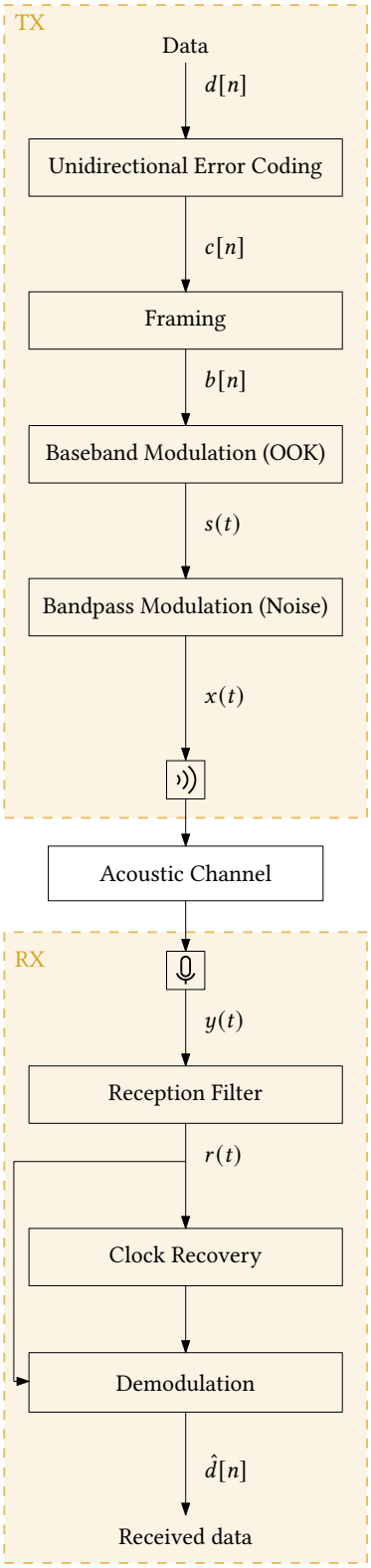
This decision function compares both slot powers of a Manchester pair with a threshold P_{th}, which influences both the robustness and security of AICs. We typically do not work with the absolute detection threshold, but with the detection threshold relative to the noise floor SNR_{th}. Security-wise, it should be as low as possible.

4 IMPLEMENTATION

In this section, we show how Acoustic Integrity Codes (AICs) and our resulting Secure Device Pairing (SDP) scheme can be implemented. We developed our system for two different platforms:

(1) **An implementation in MATLAB** for simulation and evaluation.
(2) **A proof-of-concept on Android smartphones** for practical experiments, using the Kotlin programming language.

Figure 4 shows an overview of both implementations. We can transmit and receive AICs using either implementation. It is possible



Example: Transmission of data `0xb`.

$d = (1, 0, 1, 1)$

For this example, Alice uses **Manchester Coding**, which performs the following mapping:

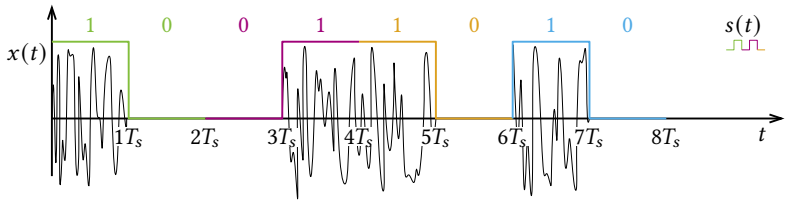
$$0 \longrightarrow 01$$
$$1 \longrightarrow 10$$

$c = (1, 0, 0, 1, 1, 0, 1, 0)$

Alice frames the data using the **delimiter** $D = (1, 1, 1, 0, 0, 0)$.

$$b = (\overbrace{1, 1, 1, 0, 0, 0}^{D}, \overbrace{1, 0, 0, 1, 1, 0, 1, 0}^{c}, \overbrace{1, 1, 1, 0, 0, 0}^{D}, \overbrace{1, 0, 0, 1, 1, 0, 1, 0}^{c}, \ldots)$$

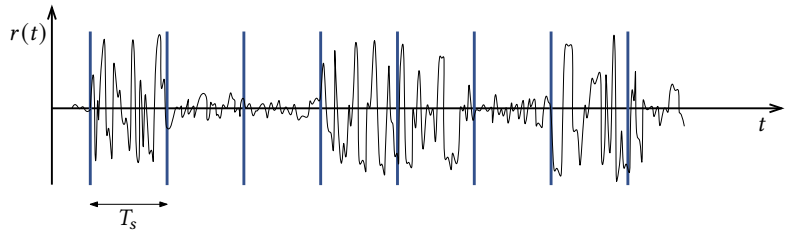
Below, we show the **baseband modulated** signal $s(t)$ as the colored signal and the **bandpass modulated** signal $x(t)$ as the grey signal. For demonstration purposes, we only plot the data part of a single frame.



Alice transmits the signal $x(t)$ using a speaker.

The channel attenuates the signal with **dampening factor** α_A and delays the signal with **propagation delay** τ_A. There is also additional **background noise** $v(t)$.

Bob receives the signal $y(t)$ using a microphone. He uses a **bandpass filter** to remove some background noise. The resulting signal is $r(t)$.



Bob determines the **frame** and **slot boundaries** based on the delimiter.

Bob measures each slot's **power** $\hat{c}_p[n]$. He applies the **decision function** $D_{ternary}$ to demodulate the data $\hat{d}[n]$ based on a **threshold** P_{th}.

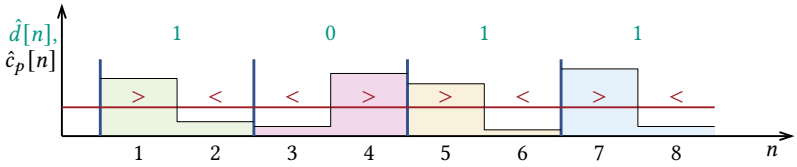


Figure 3: Construction, transmission, reception, and decoding of AIC signals, including a complete example.

to record the signals in the Android prototype into a WAV file and analyze this using MATLAB.

4.1 Simulation

We use MATLAB version 9.4 R2018a to simulate AICs. Our implementation can generate, transmit, receive, and demodulate AIC signals using the computer's speaker and microphone. Alternatively, we can also simulate a transmission using an *additive white Gaussian noise (AWGN)* channel. We implement all steps shown in Figure 3.

Figure 5 shows the spectrogram of the signal $y(t)$, which visualizes the power per frequency over time. In this example, the AIC signals' energy is concentrated in the frequency band [16 kHz, 20 kHz] with a SNR of 14 dB. We can clearly identify the three delimiters (the wider rectangles at the beginning, middle, and end), and the on and off slots in between.

4.2 Proof of Concept

Our proof-of-concept implementation runs on Android devices and is available as open-source software [32]. We can receive and transmit AIC signals, or we can record a WAV file for later analysis. We use the Android software development kit (SDK) version 28 and the Kotlin programming language version 1.3 [20] to write Android applications. We generate Java 8 compatible bytecode, which the Android SDK translates to Dalvik bytecode for use on the Android runtime (ART) on Android devices. Our Android application requires a minimum Android API level of 21, meaning that it supports all devices with Android 5.0 (released in 2014) or higher. This is not a limitation of our design, but allows for easier development of a prototype by accessing more API features.

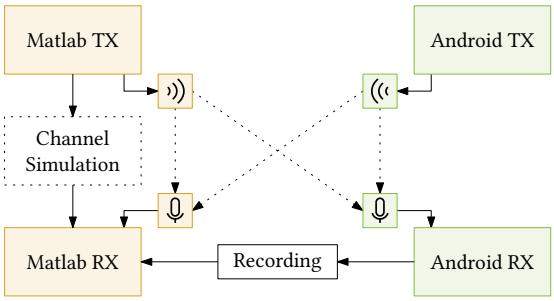


Figure 4: Overview of MATLAB and Android implementations.

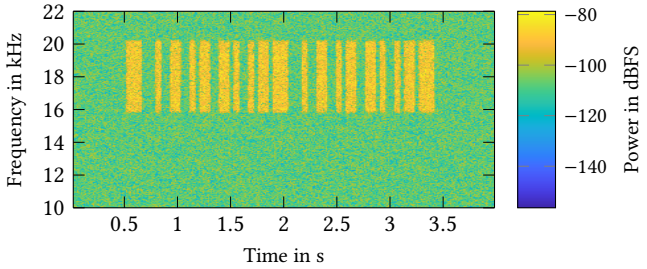


Figure 5: Spectrogram of the AIC signal $y(t)$. The frequency resolution is 50 Hz.

Our application consists of two components: (1) An Android library that handles the modulation, transmission, demodulation, and reception of AIC signals, and (2) an Android module for the user interface, which imports the library. We generate AIC signals by filling a sample buffer with Gaussian distributed random numbers for each on slot and applying a bandpass filter. We transmit this AIC signal using the Android API `android.audio.AudioTrack` in streaming mode. For reception we use the audio processing pipeline from *TarsosDSP*, which is a Java framework for real-time audio analysis [38, 39]. Figure 6 shows the Android prototype's user interface.

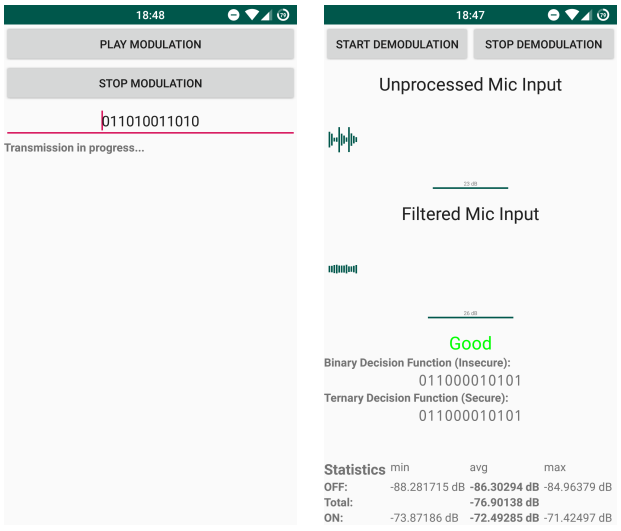
We tested our implementation on various Android smartphones: Huawei Nexus 6P (Android 8.1.0), LG G4 (Android 8.1.0), LG G5 (Android 7.1.2), LG Nexus 5 (Android 7.1), OnePlus 3T (Android 7.1.2), Samsung Galaxy S4 Mini (Android 9), Samsung Galaxy S6 (Android 7.1.2), Xiaomi Redmi K20 Pro (Android 10).

5 SECURITY ANALYSIS

In this section, we analyze if regular Integrity Codes and AICs satisfy our security goal of providing *message authentication*. A passive attack, where Mallory only eavesdrops on the communication between A and B, does not impact this security goal. The adversary can perform active attacks by sending signals, which B receives as $m'(t) = \sum_n m'_n(t)$ as part of $y(t)$ according to Equation 1. B requires a high SNR to decode the signal:

$$\mathrm{SNR} = \frac{P_{\text{signal}}}{P_{\text{noise}}} = \frac{\int_0^T (x'(t) + m'(t))^2 \, \mathrm{d}t}{\int_0^T v(t)^2 \, \mathrm{d}t}. \tag{4}$$

First, Mallory can try to disable communication between A and B via stateless jamming, which reduces the SNR by increasing P_{noise}.



(a) Transmission **(b) Reception**

Figure 6: Main views of the Android prototype.

Second, Mallory can perform *signal cancellation*[1] attacks, by sending signals that destructively interfere with the legitimate signal and, thus, preventing B from successfully decoding A's signal. This attack reduces the SNR by decreasing P_{signal}. A special case of this attack is *bit flipping*, where Mallory also sends her own message in addition to signal cancellation. Third, Mallory can perform *overshadowing* attacks by sending her signals with a power much higher than A, so that in the superposition at B the decoding process will be mostly influenced by Mallory.

We do not consider stateless jamming attacks, since protecting the availability of our system is not our security goal. As our system does not rely on the audibility of Mallory's signals, it is also not vulnerable to inaudible attacks [35, 41]. This leaves us with two distinct attack vectors on the message's integrity: signal cancellation and overshadowing.

5.1 Signal Cancellation Attacks

Integrity Codes rely on the assumption that signal cancellation is not possible [3], i.e., that it is impossible to perform a bit flip $1 \rightarrow 0$ in a signal modulated using Integrity Codes. Under this assumption, any other modification $0 \rightarrow 1$ can be detected via unidirectional error coding, which protects the integrity of the message. Signal cancellation attacks have recently gained interest in the research community [9, 26, 29]. Figure 7 shows an exemplary signal cancellation attack. Mallory uses device M_{SC} (see adversary model in Figure 2) to send a cancellation signal $m_{SC}(t)$, which minimizes the power P_{signal} received at B:

$$P_{\text{signal}} = \frac{1}{T}\int_0^T \left(x'(t) + m'_{SC}(t)\right)^2 \,\mathrm{d}t. \tag{5}$$

Destructive interference means that two waves with opposite polarity superimpose [34, pp. 212-213]. For sound pressure waves, this is also known as *active noise cancellation (ANC)*.

For the purpose of blocking communication by reducing the SNR at the receiver, signal cancellation is more challenging to perform compared to jamming, since it requires the attacker to both:

(1) predict her channel $H_{M_{SC}\rightarrow B}$ to B, and
(2) predict the signal $x'(t)$ from A at B's microphone.

Mallory can then generate a signal $m_{SC}(t)$, which destructively interferes and cancels A's signal $x'(t)$. Practical signal cancellation attacks have been demonstrated in lab environments [26, 29]. These attacks are challenging to perform and require precise synchronization, especially when canceling high-frequency signals. In the following security analysis, we assume a best case scenario for the attacker, where she is able to completely predict all channels. We use the approximate channel model from section 3 with constant attenuation α and group delay τ.

Mallory's goal is to construct a cancellation signal

$$m'_{SC}(t) = -x'(t - \tau_M) \tag{6}$$

with minimal *cancellation delay* τ_M, which requires predicting $x'(t)$. Moser et al. demonstrated a practical signal cancellation attack on predictable GPS signals in a lab environment [26]. Mallory, however, cannot *directly* predict $x'(t)$ a priori without observing A's signal $x(t)$, since $x(t)$ is a stochastic signal.

[1] Signal cancellation is also known as *signal annihilation*.

We assume that Mallory can only *indirectly* predict $x'(t)$ a posteriori using her received version $x_M(t)$ of $x(t)$. She therefore uses past values of $x_M(t)$ to predict future values of $x'(t)$. Pöpper et al. demonstrated such a signal cancellation attack on QPSK signals in a static lab environment [29]. They used two directional antennas as relays (*relaying attacker*) and relayed A's signal $x_M(t)$ to B.

We now formalize this attack. Mallory wants to relay the signal

$$x_M(t) = H_{A\rightarrow M_{SC}}\{x(t)\} = \alpha_1 x(t - \tau_1). \tag{7}$$

The channel $H_{A\rightarrow M_{SC}}$ delays the signal $x(t)$ by τ_1. When Mallory relays this signal, the channel $H_{M_{SC}\rightarrow B}$ will delay it again by τ_2. She cannot invert these delays in real-time, since she does not have access to future values of $x_M(t)$. She can only invert the attenuation of these channels. She sends the signal

$$m_{SC}(t) = -\frac{\alpha_A}{\alpha_1\alpha_2} x_M(t - \tau_r), \tag{8}$$

where $\tau_r \geq 0\,\text{s}$ is an additional delay that she can freely adjust to achieve better signal cancellation. B then receives

$$\begin{aligned} m'_{SC}(t) &= H_{M_{SC}\rightarrow B}\{m_{SC}(t)\} \\ &= -\frac{\alpha_A}{\alpha_1} x_M(t - \tau_2 - \tau_r) \\ &= -\alpha_A x(t - \tau_1 - \tau_2 - \tau_r). \end{aligned} \tag{9}$$

We can rewrite this using Equation 2 as

$$m'_{SC}(t) = -x'(t + \tau_A - \tau_1 - \tau_2 - \tau_r). \tag{10}$$

For indirect prediction of $x'(t)$, the cancellation delay (according to Equation 6) therefore is

$$\tau_M = \tau_1 + \tau_2 + \tau_r - \tau_A. \tag{11}$$

We now analyze whether a relay attack is possible against AIC signals. Since we require Mallory to operate outside the safe area S, we can give a lower bound for the cancellation delay due to the propagation delay:

$$\tau_M = \tau_r + \frac{\overline{AM_{SC}B} - \overline{AB}}{c_s} \geq \frac{2r - \overline{AB}}{c_s} \tag{12}$$

where c_s is the speed of sound in the transmission medium. The cancellation delay τ_M increases for larger safe area radii r and for smaller $\overline{AB}$. For example, Mallory can achieve $\tau_M > 1\,\text{ms}$ for realistic parameters $r > 40\,\text{cm} > \overline{AB}$. The sound wave's speed is

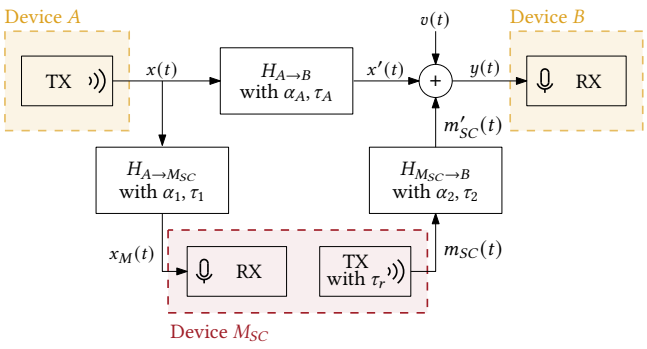


Figure 7: Adversarial device M_{SC} performing a signal cancellation attack.

an inherent physical-layer limitation that passively assists us with authentication. A related strategy is distance bounding [2], which actively measures the propagation delay and therefore requires more sophisticated implementations.

We measure the effect of this attack using the resulting power

$$P_{signal} = \frac{1}{T}\int_0^T \left(x'(t) - x'(t-\tau_M)\right)^2 \,\mathrm{d}t \tag{13}$$
$$= 2P_{x'} - 2\frac{1}{T}\int_0^T x'(t)x'(t-\tau_M)\,\mathrm{d}t.$$

Note that without any cancellation delay ($\tau_M = 0$), Mallory would be able to completely cancel the signal $x'(t)$. Otherwise, Mallory has to *maximize* the subtrahend, which contains the autocorrelation of the signal $x'(t)$:

$$R_{x'x'}(\tau_M) = \int x'(t)x'(t-\tau_M)\,\mathrm{d}t. \tag{14}$$

This is related to the autocorrelation of the original signal $x(t)$:

$$R_{x'x'}(\tau_M) = \alpha_A^2 \int x(t-\tau_A)x(t-\tau_A-\tau_M)\,\mathrm{d}t = \alpha_A^2 R_{xx}(\tau_M). \tag{15}$$

To defend against relay attacks, we therefore have to construct AIC signals with *low autocorrelation* $R_{xx}(\tau_M) \approx 0$ for $\tau_M > 1\,\mathrm{ms}$. The on slots in AIC signals consist of a *stochastic "carrier" signal* $w(t)$, which we generate by sampling from the stochastic process $\{W(t)\}$. We implement AICs using Gaussian distributed signals $\{W_{WGN}(t)\}$ for the on slots, which is optimal because white gaussian noise has minimal autocorrelation [31, p. 189].

To simplify the implementation, most publications on Integrity Codes use an existing modulation scheme with random symbols as the on slots: FSK [3], QPSK [16], or OFDM in combination with QAM [3, 10, 27]. This simplifies the implementation because parts of an existing physical-layer pipeline, such as a Wi-Fi chip or an SDR reference implementation, can be reused. It is a security tradeoff, though, since these modulation schemes usually have high autocorrelation, depending on the slot size T_S, which in turn is prone to signal cancellation.

5.2 Evaluation of Signal Cancellation Attacks

We compare two choices of $\{W(t)\}$ using MATLAB simulations:

(1) On slots containing *QPSK signals*, which have high autocorrelation. Most other implementations of Integrity Codes use an existing modulation scheme such as QPSK.
(2) On slots containing *Gaussian distributed signals*, which have low autocorrelation. This corresponds to our implementation of AICs.

For both cases, we measure the autocorrelation coefficient and the attenuation that a signal cancellation attacker achieves for different cancellation delays τ_M. Our evaluation applies to different device distances $\overline{AB}$ and attacker locations according to Equation 12. Mallory aims to achieve high attenuation to cancel Alice's signal. We do not vary the SNR, because we assume a best-case scenario for Mallory where she is able to match Alice's SNR. We use the frequency band $[200\,\mathrm{Hz}, 800\,\mathrm{Hz}]$ to better visualize the security impact. For higher frequencies, signal cancellation is even more challenging due to stricter timing constraints.

5.2.1 QPSK On Slots. Figure 8a shows an example of an AIC signal $x(t)$ using the random process $\{W_{QPSK}(t)\}$, which is a non-stationary random process containing random QPSK symbols (drawn independently with uniform probability). For this example, each QPSK symbol has duration $T_Q = \frac{T_S}{4}$, such that each slot contains four QPSK symbols. This use of *"minislots"* increases the security [3], by reducing the autocorrelation. We use a gross bit rate $R_g \approx 21.8\,\mathrm{bps}$ and a carrier frequency $f_c = 500\,\mathrm{Hz}$.

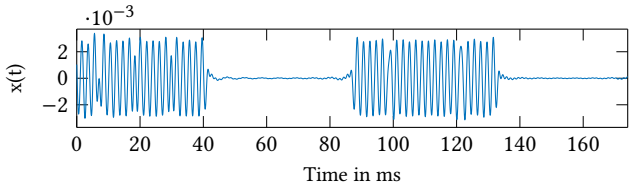
Even though the content of each on slot is "random" in the sense that it consists of four QPSK symbols, where each symbol was independently drawn from one of four possible QPSK symbols, it is still deterministic during each of these QPSK symbols. The underlying period f_c of the deterministic carrier signal can be clearly seen. Each QPSK symbol only carries two bits of information.

Figure 9a shows the attenuation that Mallory achieves for different time delays using a relay attack. Destructive interference occurs at multiples of the carrier period $\frac{1}{f_c} = 2\,\mathrm{ms}$, which is possible even for realistic values of $\tau_M > 1\,\mathrm{ms}$ (dashed line). For most time delays and for high bit rates, however, the signals interfere constructively and the attenuation is $< 0\,\mathrm{dB}$. Mallory therefore has to precisely control her additional delay τ_r in Equation 11.

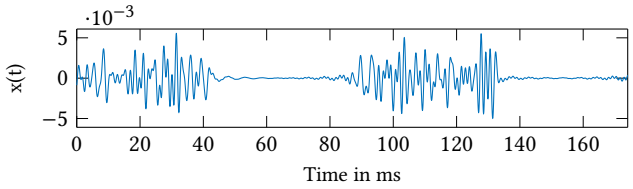
Figure 9b shows the autocorrelation $R_{xx}(\tau_M)$ of an AIC signal using $\{W_{QPSK}(t)\}$. The autocorrelation has the same peaks as the attenuation in Figure 9a, at multiples of the carrier period $\frac{1}{f_c} = 2\,\mathrm{ms}$. This is consistent with our argument that signal cancellation requires high autocorrelation.

5.2.2 Gaussian On Slots. Figure 8b shows an example of an AIC signal $x(t)$ using a white Gaussian noise process $\{W_{WGN}(t)\}$. Compared to the QPSK-shaped on slots, the Gaussian-shaped on slots carry more information and are therefore harder to predict. We cannot determine any obvious patterns when looking at the plot.

Figure 10a shows the attenuation that Mallory achieves for different time delays using a relay attack. The attenuation is positive only for $\tau_M \approx 0$, where Mallory can successfully cancel the AIC signal. Figure 10b shows the corresponding autocorrelation $R_{xx}(\tau_M)$ of



(a) QPSK on slots.



(b) WGN on slots.

Figure 8: Four slots of an AIC signal using different on slot implementations.

the AIC signal. The autocorrelation coefficient is approximately zero for $\tau_M > 1.5$ ms. When using a higher frequency band, such as [16 kHz, 20 kHz], Mallory needs to achieve even lower cancellation delays $\tau_M \ll 1$ ms. As this is not possible for realistic safe areas, Mallory's cancellation signal actually increases the received power.

5.3 Overshadowing Attacks

In overshadowing attacks, Mallory attempts to send her own AIC signal $m_O(t)$ (see Figure 2) with much greater power than the legitimate signal $x(t)$, so that her signal $m_O(t)$ determines the data that B decodes. In contrast to signal cancellation attacks, Mallory does not necessarily need to receive the legitimate signal. She could use it, however, to obtain timing information and synchronize with the legitimate signal.

We defend against overshadowing attacks by adjusting the receiver's detection step. The receiver measures the power $\hat{c}_p[n]$ of every slot and applies a decision function $D : \mathbb{R} \times \mathbb{R} \mapsto S$ on every Manchester pair to determine the bit $\hat{d}[n]$ that this pair encodes:

$$\hat{d}[n] = D\left(\hat{c}_p[2n], \hat{c}_p[2n+1]\right). \tag{16}$$

Overshadowing attacks result in high slot powers in both slots, which our decision function D_{ternary} (see Equation 3) detects due to the threshold P_{th}. The threshold should be as low as possible but definitely lower than Alice's average signal power. If an overshadowing attacker attempts a bit flip, her off slot overlaps with the legitimate on slot, which has higher power than P_{th}. For conventional modulation schemes such as ASK, however, existing decoders do not consider an attacker at the physical layer and instead aim at maximizing robustness to increase throughput. These binary decision functions D_{binary} decode the Manchester encoded slots based on a relative comparison between the slot powers, which is vulnerable to overshadowing attacks.

Table 1: Comparison of D_{binary} and D_{ternary} in the presence of an overshadowing attacker transmitting d_M instead of the legitimate d. The highlighted rows indicate attempted bit flips.

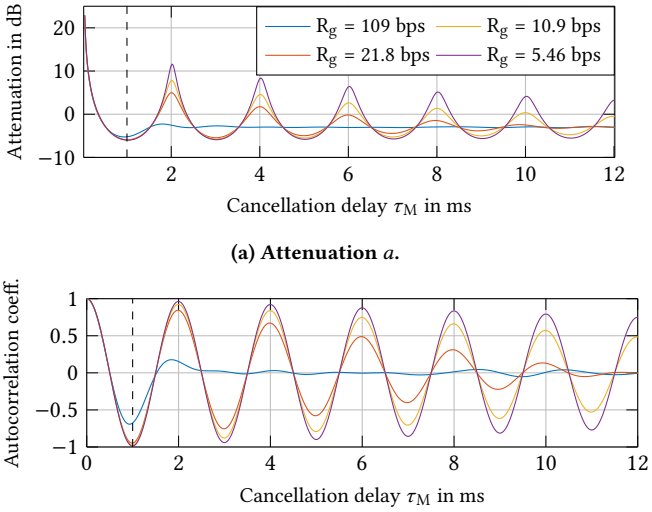
d	d_M	p_1/dBFS	p_2/dBFS	$D_{\text{binary}}(p_1, p_2)$	$D_{\text{ternary}}(p_1, p_2)$
0	-	-90	-70	0	0
0	0	-90	-59.6	0	0
0	1	-60	-70	1	ϵ
1	-	-70	-90	1	1
1	1	-59.6	-90	1	1
1	0	-70	-60	0	ϵ

5.4 Evaluation of Overshadowing Attacks

We consider an overshadowing attacker in an exemplary scenario with a noise floor of −90 dBFS, an SNR of 20 dB for the legitimate sender, a higher SNR of 30 dB for the overshadowing attacker, and a threshold of −80 dBFS at the receiver. Table 1 lists all combinations of legitimate message bit d and adversarial message bit d_M. The receiver can detect bit flips (highlighted rows) using the secure decision function D_{ternary}, while D_{binary} is vulnerable.

6 ROBUSTNESS EVALUATION

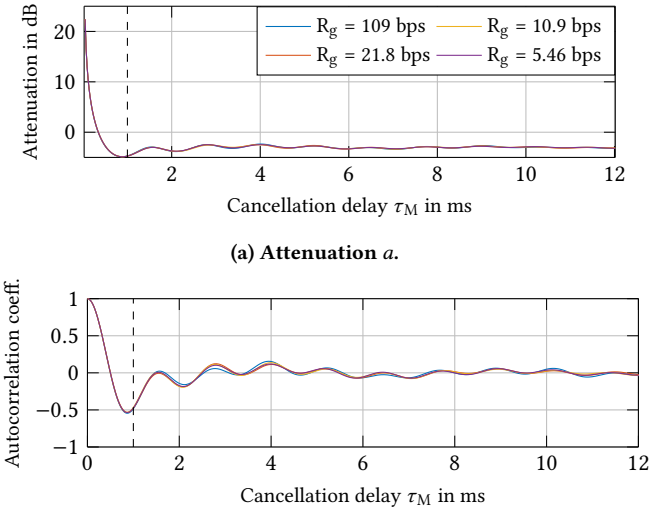
In the last section, we considered an active adversary attacking our system. We now evaluate the robustness of AICs when there is no active attacker. In this case, we have to deal with background noise on the acoustic channel.



(a) Attenuation a.

(b) Autocorrelation $R_{xx}(\tau_M)$.

Figure 9: Attenuation and Autocorrelation of an AIC signal using $\{W_{QPSK}(t)\}$.



(a) Attenuation a.

(b) Autocorrelation $R_{xx}(\tau_M)$.

Figure 10: Attenuation and Autocorrelation of an AIC signal using $\{W_{WGN}(t)\}$.

6.1 Methodology

We use our MATLAB implementation to simulate AIC signals and background noise. We vary the signal-to-noise ratio (SNR) at the receiver (see Equation 4), the gross bit rate $R_g = \frac{1}{T_s}$, the detection threshold $\mathrm{SNR}_{\mathrm{th}} = \frac{P_{\mathrm{th}}}{P_{\mathrm{Noise}}}$, and the bandwidth of our system. Our evaluation results do not depend on the transmit power or the device distance $\overline{AB}$, as these parameters both influence the resulting SNR. We measure the resulting bit error ratio (BER).

All simulations use a sample rate of $f_s = 44.1\,\mathrm{kHz}$. We simulate additive white Gaussian noise with a noise power of $-87\,\mathrm{dBFS}$ in the frequency band. We transmit 128 bits of random data d. We repeat all experiments 200 times and give the average result.

6.2 Inter-Symbol Interference

The bandwidth influences how fast the transitions between on slots and off slots can happen. For low bandwidths, this transition takes longer and the on slots' energy spreads to the off slots, which is called *inter-symbol interference (ISI)*. We observe that both very low and very high SNRs suffer from low bandwidth. This occurs for different reasons:

- For *low signal powers* below the detection threshold, the limited bandwidth further reduces the power in the on slots, leading to detection difficulties at the receiver.
- For *high signal powers* well above the detection threshold, the limited bandwidth leads to increased power in the off slots due to ISI. On slots and off slots are then above the detection threshold, which is rejected by the decision function.

6.3 Results

We achieve low BERs below 1 % for $R_g < 450\,\mathrm{bps}$ and $\mathrm{SNR} > \mathrm{SNR}_{\mathrm{th}} = 11\,\mathrm{dB}$ using bandwidths larger than 4 kHz. For a net bit rate of 100 bps, e.g., we achieve a BER below 0.1 % at an SNR of 14 dB. This corresponds to a transmission time of approximately 1.2 s for a 128 bit hash value. The BER increases for higher bit rates or lower bandwidths due to ISI. Using the insecure decision function D_{binary} improves robustness at the cost of being vulnerable to overshadowing attacks (see subsection 5.3).

The detection threshold $\mathrm{SNR}_{\mathrm{th}}$ influences the robustness of our pairing scheme using the ternary decision function D_{ternary}. In Figure 11, we plot the BER for different SNRs and detection thresholds, using the gross bit rate $R_g = 220\,\mathrm{bps}$. For a fixed SNR, the BER first decreases to approximately 0 % and then increases again to 100 %:

- For $\mathrm{SNR}_{\mathrm{th}} \approx 0\,\mathrm{dB}$, ISI leads to bit errors because the on slots' energy spreads to the off slots, which both surpass the detection threshold. This is why high SNRs, where the on slots contain more energy, perform worse in the left half of the plot.
- The increase in the right half of the plot is due to the detection threshold surpassing the power of the on slots. This occurs at ca. $\mathrm{SNR}_{\mathrm{th}} \approx \mathrm{SNR} + 3\,\mathrm{dB}$ (dashed lines), due to the on slots having approximately twice the power than the average signal power SNR.

The ISI in the plot's left half decreases for lower bit rates. Our results indicate that there is less tolerance on the detection threshold for high bit rates and varying SNR values. For low bit rates, the range of detection thresholds that allow error-free transmission increases.

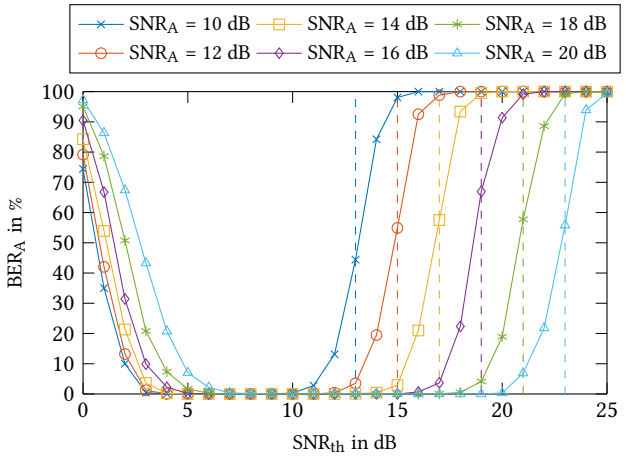


Figure 11: BER for different detection thresholds $\mathrm{SNR}_{\mathrm{th}}$ and SNRs.

7 CONCLUSION

SDP relies on an OOB channel to authenticate devices. We designed an SDP scheme using short-range acoustic communication to transmit key material. We proposed *Acoustic Integrity Codes (AICs)* to achieve message authentication on the acoustic physical layer. To the best of our knowledge, Integrity Codes have not been used to secure acoustic communication before.

Integrity Codes can be vulnerable to signal cancellation attacks if the transmitted on slots are not sufficiently random. Our security analysis shows that we can defend against *signal cancellation attacks* by designing signals with low autocorrelation, e.g., Gaussian distributed signals. We introduced a set of realistic operation parameters that mitigate signal cancellation attacks via additional propagation delays: Compared to conventional modulation schemes that do not consider an attacker at the physical layer, our system can also detect *overshadowing attacks* by using a threshold in the receiver's decision function. The attacker cannot impersonate the legitimate sender.

The robustness of AICs depends on the channel conditions and the desired bit rate. Our evaluation demonstrated that lower bit rates, higher signal-to-noise ratios (SNRs), and higher bandwidths improve the bit error rate. Finally, we implemented a *proof-of-concept* for Android devices to demonstrate practical pairing between different smartphone models.

ACKNOWLEDGMENTS

This work has been co-funded by the LOEWE initiative (Hessen State Ministry for Higher Education, Research and the Arts, Germany) within the emergenCITY centre and by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

REFERENCES

[1] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. 2002. Talking to Strangers: Authentication in Ad-Hoc Wireless Networks. In *NDSS*.

[2] S. Brands and D. Chaum. 1994. Distance-Bounding Protocols. In *Advances in Cryptology — EUROCRYPT '93*. Springer Berlin Heidelberg, Berlin, Heidelberg, 344–359. ISBN: 978-3-540-48285-7.

[3] S. Čapkun, M. Čagalj, R. Rengaswamy, I. Tsigkogiannis, J. Hubaux, and M. Srivastava. 2008. Integrity Codes: Message Integrity Protection and Authentication over Insecure Channels. *IEEE Transactions on Dependable and Secure Computing*, 5, 4, (Oct. 2008), 208–223. ISSN: 1545-5971. DOI: 10.1109/TDSC.2008.11.

[4] W. R. Claycomb and D. Shin. 2009. Secure device pairing using audio. In *43rd Annual 2009 International Carnahan Conference on Security Technology*. (Oct. 2009), 77–84. DOI: 10.1109/CCST.2009.5335562.

[5] D. Dolev and A. Yao. 1983. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29, 2, (Mar. 1983), 198–208. ISSN: 0018-9448. DOI: 10.1109/TIT.1983.1056650.

[6] N. Ferguson, B. Schneier, and T. Kohno. 2010. *Cryptography Engineering*. Wiley Publishing, Inc.

[7] M. Fomichev, F. Alvarez, D. Steinmetzer, P. Gardner-Stephen, and M. Hollick. 2018. Survey and Systematization of Secure Device Pairing. *IEEE Communications Surveys & Tutorials*, 20, 1, 517–550. DOI: 10.1109/comst.2017.2748278.

[8] M. Fomichev, M. Maass, L. Almon, A. Molina, and M. Hollick. 2019. Perils of Zero-Interaction Security in the Internet of Things. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3, 1, (Mar. 2019), 1–38. ISSN: 2474-9567. DOI: 10.1145/3314397.

[9] N. Ghose, L. Lazos, and M. Li. 2018. Secure Device Bootstrapping Without Secrets Resistant to Signal Manipulation Attacks. In *2018 IEEE Symposium on Security and Privacy (SP)*. (May 2018), 819–835. DOI: 10.1109/SP.2018.00055.

[10] S. Gollakota, N. Ahmed, N. Zeldovich, and D. Katabi. 2011. Secure In-band Wireless Pairing. In *Proceedings of the 20th USENIX Conference on Security* (SEC'11). USENIX Association, San Francisco, CA, 16–16. http://dl.acm.org/citation.cfm?id=2028067.2028083.

[11] M. T. Goodrich, M. Sirivianos, J. Solis, C. Soriente, G. Tsudik, and E. Uzun. 2009. Using Audio in Secure Device Pairing. *Int. J. Secur. Netw.*, 4, 1/2, (Feb. 2009), 57–68. ISSN: 1747-8405. DOI: 10.1504/IJSN.2009.023426.

[12] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. 2006. Loud and Clear: Human-Verifiable Authentication Based on Audio. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS'06)*. IEEE. DOI: 10.1109/icdcs.2006.52.

[13] T. Halevi and N. Saxena. 2010. On Pairing Constrained Wireless Devices Based on Secrecy of Auxiliary Channels: The Case of Acoustic Eavesdropping. In *Proceedings of the 17th ACM Conference on Computer and Communications Security* (CCS '10). ACM, Chicago, Illinois, USA, 97–108. ISBN: 978-1-4503-0245-6. DOI: 10.1145/1866307.1866319.

[14] D. Halperin, T. S. Heydt-Benjamin, B. Ransford, S. S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. H. Maisel. 2008. Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*. (May 2008), 129–142. DOI: 10.1109/SP.2008.31.

[15] J. Han, Y.-H. Lin, A. Perrig, and F. Bai. 2014. Short Paper: MVSec: Secure and Easy-to-use Pairing of Mobile Devices with Vehicles. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks* (WiSec '14). ACM, Oxford, United Kingdom, 51–56. ISBN: 978-1-4503-2972-9. DOI: 10.1145/2627393.2627400.

[16] Y. Hou, M. Li, R. Chauhan, R. M. Gerdes, and K. Zeng. 2015. Message Integrity Protection over Wireless Channel by Countering Signal Cancellation: Theory and Practice. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security* (ASIA CCS '15). ACM, Singapore, Republic of Singapore, 261–272. ISBN: 978-1-4503-3245-3. DOI: 10.1145/2714576.2714617.

[17] Q. Hu, J. Zhang, A. Mitrokotsa, and G. Hancke. 2018. Tangible Security: Survey of Methods Supporting Secure Ad-Hoc Connects of Edge Devices with Physical Context. *Computers & Security*, 78, (Sept. 2018), 281–300. DOI: 10.1016/j.cose.2018.06.009.

[18] J. Katz, A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. 1996. *Handbook of Applied Cryptography*. CRC press.

[19] T. Kindberg and K. Zhang. 2003. Validating and Securing Spontaneous Associations between Wireless Devices. In *Information Security*. Springer Berlin Heidelberg, Berlin, Heidelberg, 44–53. ISBN: 978-3-540-39981-0.

[20] Kotlin Foundation. 2019. Kotlin Programming Language. Retrieved Nov. 25, 2019 from https://kotlinlang.org/.

[21] C. V. Lopes and P. M. Q. Aguiar. 2001. Aerial acoustic communications. In *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*. (Oct. 2001), 219–222. DOI: 10.1109/ASPAA.2001.969582.

[22] R. Mayrhofer, J. Fuß, and I. Ion. 2013. UACAP: A Unified Auxiliary Channel Authentication Protocol. *IEEE Transactions on Mobile Computing*, 12, 4, (Apr. 2013), 710–721. DOI: 10.1109/tmc.2012.43.

[23] R. Mayrhofer, H. Gellersen, and M. Hazas. 2006. An Authentication Protocol using Ultrasonic Ranging. Tech. rep. COMP-002-2006. Lancaster University, (Oct. 2006).

[24] R. Mayrhofer, H. Gellersen, and M. Hazas. 2007. Security by Spatial Reference: Using Relative Positioning to Authenticate Devices for Spontaneous Interaction. In *UbiComp 2007: Ubiquitous Computing*. Springer Berlin Heidelberg, Berlin, Heidelberg, 199–216. ISBN: 978-3-540-74853-3.

[25] M. Miettinen, N. Asokan, T. D. Nguyen, A.-R. Sadeghi, and M. Sobhani. 2014. Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (CCS '14). ACM, Scottsdale, Arizona, USA, 880–891. ISBN: 978-1-4503-2957-6. DOI: 10.1145/2660267.2660334.

[26] D. Moser, V. Lenders, and S. Capkun. 2019. Digital Radio Signal Cancellation Attacks: An Experimental Evaluation. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks* (WiSec '19). ACM, Miami, Florida, 23–33. ISBN: 978-1-4503-6726-4. DOI: 10.1145/3317549.3319720.

[27] Y. Pan, Y. Hou, M. Li, R. M. Gerdes, K. Zeng, M. A. Towfiq, and B. A. Cetiner. 2017. Message Integrity Protection over Wireless Channel: Countering Signal Cancellation via Channel Randomization. *IEEE Transactions on Dependable and Secure Computing*, 1–1. DOI: 10.1109/tdsc.2017.2751600.

[28] C. Peng, G. Shen, Y. Zhang, and S. Lu. 2009. Point&Connect: Intention-based Device Pairing for Mobile Phone Users. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services* (MobiSys '09). ACM, Kraków, Poland, 137–150. ISBN: 978-1-60558-566-6. DOI: 10.1145/1555816.1555831.

[29] C. Pöpper, N. O. Tippenhauer, B. Danev, and S. Capkun. 2011. Investigation of Signal and Message Manipulations on the Wireless Channel. In *Computer Security – ESORICS 2011*. Springer Berlin Heidelberg, Berlin, Heidelberg, 40–59. ISBN: 978-3-642-23822-2.

[30] R. Prasad and N. Saxena. 2008. Efficient Device Pairing Using "Human-Comparable" Synchronized Audiovisual Patterns. In *Applied Cryptography and Network Security*. Springer Berlin Heidelberg, Berlin, Heidelberg, 328–345. ISBN: 978-3-540-68914-0. DOI: https://doi.org/10.1007/978-3-540-68914-0_20.

[31] J. G. Proakis. 2002. *Communication Systems Engineering*. (2nd ed.). Prentice Hall, Upper Saddle River, N.J. ISBN: 0130617938.

[32] F. Putz. 2020. Acoustic Integrity Codes Android Prototype. https://seemoo.de/aic-prototype.

[33] Q. Quach, N. Nguyen, and T. Dinh. 2014. Secure Authentication for Mobile Devices Based on Acoustic Background Fingerprint. In *Advances in Intelligent Systems and Computing*. Springer International Publishing, 375–387. DOI: 10.1007/978-3-319-02741-8_32.

[34] T. D. Rossing, (Ed.) 2007. *Springer Handbook of Acoustics*. Springer.

[35] N. Roy, S. Shen, H. Hassanieh, and R. R. Choudhury. 2018. Inaudible Voice Commands: The Long-Range Attack and Defense. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Renton, WA, (Apr. 2018), 547–560. ISBN: 978-1-939133-01-4.

[36] D. Schürmann and S. Sigg. 2013. Secure Communication Based on Ambient Audio. *IEEE Transactions on Mobile Computing*, 12, 2, (Feb. 2013), 358–370. DOI: 10.1109/tmc.2011.271.

[37] W. Shen, B. Yin, L. Liu, X. Cao, Y. Cheng, Q. Li, and W. Wang. 2016. Secure In-Band Bootstrapping for Wireless Personal Area Networks. *IEEE Internet of Things Journal*, 3, 6, (Dec. 2016), 1385–1394. DOI: 10.1109/jiot.2016.2604221.

[38] J. Six. 2015. *Digital Sound Processing and Java. Documentation for the TarsosDSP Audio Processing Library*. (2.3 ed.). IPEM, Ghent University. (May 2015).

[39] J. Six, O. Cornelis, and M. Leman. 2014. TarsosDSP, a Real-Time Audio Processing Framework in Java. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. (Jan. 2014).

[40] C. Soriente, G. Tsudik, and E Uzun. 2008. HAPADEP: Human-Assisted Pure Audio Device Pairing. In *Information Security*. Springer Berlin Heidelberg, Berlin, Heidelberg, 385–400. DOI: 10.1007/978-3-540-85886-7_27.

[41] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu. 2017. DolphinAttack: Inaudible Voice Commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (CCS '17). ACM, Dallas, Texas, USA, 103–117. ISBN: 978-1-4503-4946-8. DOI: 10.1145/3133956.3134052.