Advantages and Risks of *Sensing* for Cyber-Physical *Security*

Submitted in partial fulfillment for the requirements for
the degreee of
Doctor of Philosophy
in
Electrical & Computer Engineering

Jun Han

B.S., Electrical & Computer Engineering, Carnegie Mellon University
M.S., Electrical & Computer Engineering, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

May 2018

I dedicate this thesis to my beloved parents, Hyun Sook Lee & Myung Woo Han.

Abstract

With the the emergence of the Internet-of-Things (IoT) and Cyber-Physical Systems (CPS), modern computing is now transforming from residing only in the cyber domain to the cyber-physical domain. I focus on one important aspect of this transformation, namely shortcomings of traditional security measures. Security research over the last couple of decades focused on protecting data in regard to identities or similar static attributes. However, in the physical world, data rely more on physical relationships, hence requires CPS to verify *identities* together with *relative physical context* to provide security guarantees. To enable such verification, it requires the devices to prove unique relative physical context only available to the intended devices. In this work, I study how varying levels of constraints on *physical boundary* of co-located devices determine the relative physical context. Specifically, I explore different application scenarios with varying levels of constraints – including smart-home, semi-autonomous vehicles, and in-vehicle environments – and analyze how different constraints affect binding identities to physical relationships, ultimately enabling IoT devices to perform such verification. Furthermore, I also demonstrate that sensing may pose risks for CPS by presenting an attack on personal privacy in a smart home environment.

Acknowledgments

I would like to express my sincere gratitude to my advisor, Professor Patrick Tague, for his advice, support, and encouragements throughout my doctorate studies. I am extremely thankful to him for helping me get through one of the most difficult times of my life. I am so grateful for the numerous meetings and discussions, and for his inspirations, patience, and encouragements. He showed me, through his actions, that he prioritizes and truly cares about his students. Furthermore, I learned how to be a better person from him, and for that I will always be grateful. Patrick, you are a true mentor and I, too, want to be an advisor like you.

I would also like to take this opportunity to thank my thesis committee members, Professor Anupam Datta, Professor Pei Zhang, and Professor Marco Gruteser. I am very grateful for their invaluable feedback, time, and help.

Furthermore, I am thankful to my wonderful collaborators and colleagues. I thank past and current members of MEWS group – Madhu, Sean, Le, Emmanuel, Yuan, Dimitrios, Mike, Yu-Seung, Bruce, and Brian – for providing such a friendly group atmosphere. I also thank my recent co-authors Albert, Manal, Shijia, Xinlei, and Carlos. I would also like to thank Dr. Fan Bai from General Motor Research for his advice and help. I want to thank my colleagues throughout my Ph.D. program – Ming, Tae-Ho, Chris, Min Suk, Tenma, Soo Bum, Jiyong, Sang Kil, Hyun Jin, Hsu-Chun, Chen, and Payas. Finally, I would like to acknowledge the generous support from the following funding sources: NSF CNS-1645759, and Carnegie Institute of Technology (*Ann & Martin McGuinn* and *Frank J. Marshall*) Graduate Fellowships.

I thank my friends – Joohoon, Yuni, Hyo-sang, Wonkyum, Won Jae, and Jae Yoon – for the wonderful times together. I would also like to thank Sukun for being a good friend and a mentor.

I am always indebted to my beloved parents, Hyun Sook Lee and Myung Woo Han, for their unconditional love, prayers, encouragements, and support. I could not have made it through this difficult and long journey without my parents. I love you Mom and Dad!

Most importantly, I want to thank my Lord and Savior, Jesus Christ, for your

grace and blessings throughout my Ph.D. journey. You are my advisor, mentor, and a friend. Thank you very much. *"So do not fear, for I am with you; do not be dismayed, for I am your God. I will strengthen you and help you; I will uphold you with my righteous right hand." Isaiah 41:10*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Traditionally, computational devices are stand-alone devices that merely process information that is input to them via input interfaces (e.g., keyboard, mouse, keypad, etc.). Hence, they have limited interactions with the surrounding environments. Desktop machines, 2G feature phones, and cars are great examples of such devices.

However, over the last decade, devices have been designed to interact with the physical world, especially with the advent of the Internet-of-Things (IoT). The old home desktop machine is now transforming into a smart home, filled with various networked computational devices, such as Amazon Echo, Smart TV, Internet-Connected Refrigerator, etc. Traditional 2G feature phones are transforming themselves into smartphones, full of sensors that monitor and interact with the outside world. Cars, too, are transforming to self-driving cars, which are all about monitoring the environment to make autonomous driving decisions. This transformation will continue to expand with the emergence of connected V2V infrastructure. The overwhelming number of sensors along with improved artificial intelligence helps devices to make autonomous decisions, which starts to resemble some traits of human beings, especially on how we sense and interact, or *perceive* the physical world.

1.1 Challenges of Cyber-Physical Security

However, we pose the following important question regarding such transformation. Is the security we have sufficient and adequate enough to cope with such transformation? I claim that it is not, and illustrate the shortcomings of the security today in this thesis. Traditional security concentrated on protecting data in regard to *identities* or similar static attributes. Decades of research in this area resulted in significant contributions such as development of

cryptographic algorithms to ensure different security properties, and utilizing the algorithms to create security tools and protocols that many of us enjoy today. For example, we utilize authentication protocols to verify certain identity using cryptographic solutions. Accessing online banking such as Bank of America website leverages SSL/TLS, which enables server authentication.

Traditionally, authentication (i.e., verification of identity) constituted of the following three factors:

- *What you know*: Password, PIN, SSN
- *What you have*: Digital Certificate, Door Key, Credit Card
- *Who you are*: Biometric (such as fingerprint, iris, face)

All of the above traditional notion of authentication require an important step to *pre-register prior knowledge* to map the *identity* to certain *attribute*. For example, creating Gmail service requires the user to create ID and Password before usage. Similarly, using FaceID or TouchID on an iPhone requires registering one's face or fringerprint information.

However, as we move into the cyber-physical world, data rely more on the *physical relationships* among the involved parties rather than the *identities alone*. Specifically, identity-based solutions (such as the crypto systems) alone are not adequate to prove the relative physical relationships. For example, when there are two cars on a road as depicted in Figure 1.1, it is difficult for the cars to verify one's location relative to another car. Specifically, if *Car M* is malicious and wants to fool *Car A* into believing that *Car M* is behind *Car A*, it is difficult for *Car A* to verify that information. Simply sending GPS information and digitally signing the message will not help because GPS information can at best be low in accuracy and is vulnerable to spoofing attacks. In addition, signatures only verify the authenticity of the message and not the physical relationships. In order to overcome this challenge, I ask the following question: *How can the IoT devices verify and bind the* ***identities*** *to their* ***physical relationships****?*

1.2 *Signals-of-Opportunity*: Contextual Information Available in Physical Environments

In order to answer the above question, I take inspirations from how we, as humans, interact with each other in our physical world and apply the inspirations to the IoT devices.

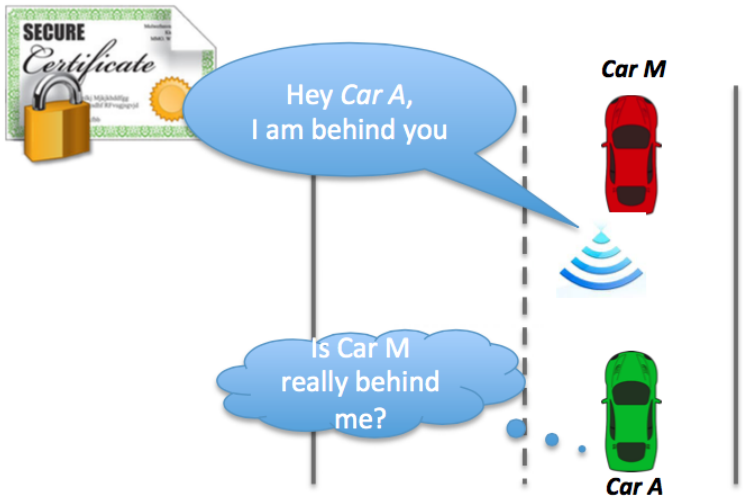


Figure 1.1: Figure depicts the difficulty of *Car A* attempting to verify the location of *Car M*. Digitally signing the GPS coordinates is not an adequate solution because GPS can easily be spoofed.

1.2.1 Inspirations from Humans

Specifically, how do we identify and verify each others' identity? We actually do not just depend on identities (i.e., names) alone, but with various other contextual cues, by sensing and interpreting sensed information. For example, when your manager at work calls your name, you first identify your manager, and implicitly verify that identity with various contextual cues including appearance, voice, intonation, vocabulary, as well as spatio-temporal information. Then you would cross-verify across these multiple context to arrive at the conclusion that it was indeed your boss calling your name.

Likewise, we also interact with our physical environments by depending on various forms of sensory perceptions as well. For example, we recognize a door opening and closing by verifying the information across different senses – we hear, see, and sometimes feel the vibration of the door opening and closing.

1.2.2 Sensors to Capture Signals-of-Opportunity

In this thesis, I am inspired by the aforementioned human behavior and propose to apply the inspirations to the IoT devices. Specifically, I enable the IoT devices and cyber-physical systems to also capture contextual information abundantly available in the physical environment, also known as *Signals-of-Opportunity* to assist with the binding. Such Signals-of-Opportunity – ranging from speakers emitting sound, fridge or dryer humming, and to vehicles experiencing

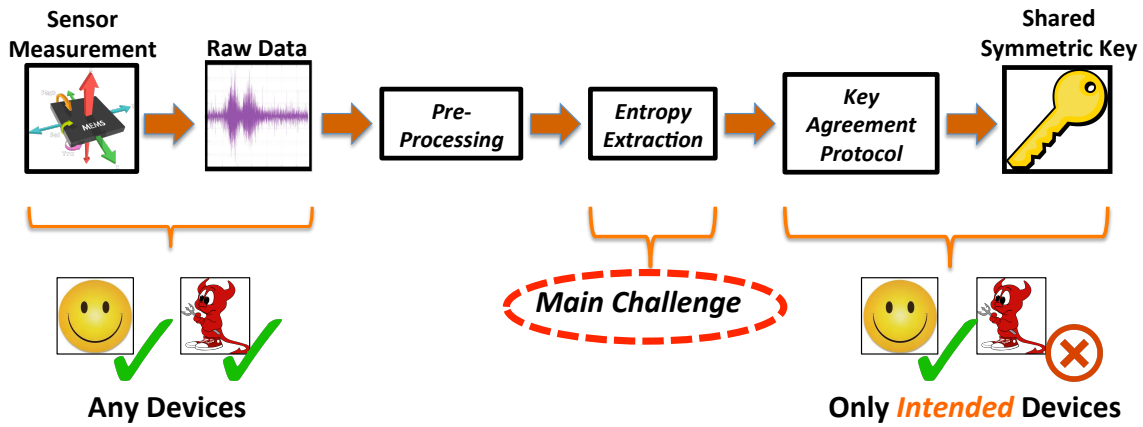


Figure 1.2: Figure depicts the generic *Context-based Pairing* framework. Main contribution of my work lies within the Entropy Extraction Module.

bumps on a road – can be captured with the sensors that are equipped in the devices. The devices then may use the sensed data to *verify the physical relationships* across devices (e.g., proving co-location of two devices) to complement security protocols.

1.3 Context-based Pairing

In order to study how devices may bind their identities to their corresponding physical relationships, I study context-based pairing. This is because context-based pairing is a process that enables two co-located devices to discover, authenticate, and create a secure channel by proving their physical relationships. The devices make use of sensor measurements to capture common contextual cues to prove their physical relationships (e.g., co-location of two cars on a road). I first present the generic framework, main challenges, and desirable properties.

1.3.1 Generic Framework

Figure 1.2 depicts the generic context-based pairing framework. Each party involved in the pairing process first leverages its sensor to measure the corresponding Signals-of-Opportunity, resulting in depicted raw signal. The signal is then pre-processed, and subsequently input to the *Entropy Extraction Module*. This module translates the pre-processed sensor signal into entropy bits that is then input to a cryptographic key agreement protocol. As I will

demonstrate from the following Chapter 1.4, how to extract entropy from the collected sensor data depends largely on the physical constraint of the application, namely the physical boundary. For example, if the two devices are located in a complete insulation (e.g., perfectly soundproof room), then the two devices would simply exchange entropy bits over audio channels. However, relaxing the constraint (hence more practical problems) increases the difficulty of such entropy extraction protocol, as the solution cannot allow potential attackers to also extract the common entropy. I note that this protocol involves communication across the involved parties over wireless communication medium such as Wi-Fi, Bluetooth, or ZigBee. At the end of the framework, the two parties arrive at a shared symmetric cryptographic key. Furthermore, capturing Signals-of-Opportunity on the left-hand side can be performed by any arbitrary devices including both legitimate and attacker devices. However, through a successful context-based pairing protocol, only the intended devices should be able to satisfy the requirements to achieve the shared symmetric key, depicted on the right-hand side. This leads us to the next question of how to enable this in the Entropy Extraction Module.

1.3.2 Main Challenges

The main challenge of context-based pairing lies in its Entropy Extraction Module in order to provide a secure binding of identities to their physical relationships. Specifically, this module needs to extract "***common***", "***secret***", "***randomness***". First, the extracted entropy needs to be "common" across the interacting devices. Hence, the entropy needs to be extracted from common source of contextual information that is commonly perceived by the interacting devices. Second, the extracted entropy needs to be "secret". Hence, the contextual information needs to be perceived only by the interacting (intended) devices. Otherwise, the attackers may launch impersonation attacks. Third, the extracted entropy needs to have sufficient "randomness". Hence the contextual information should be unpredictable. Otherwise, the attacker may again be able to guess the randomness and attack the pairing process. For example, contextual information such as door opening events may provide randomness because it is very difficult to predict the exact timing of each door opening events.

1.3.3 Desirable Properties

Taking these challenges, there are a number of desirable properties such as *scalability*, *usability*, and *performance*. First, it would be more desirable to have a *scalable* solution by reducing assumptions of hardware used. This is enable the use of available, cost-effective, and/or

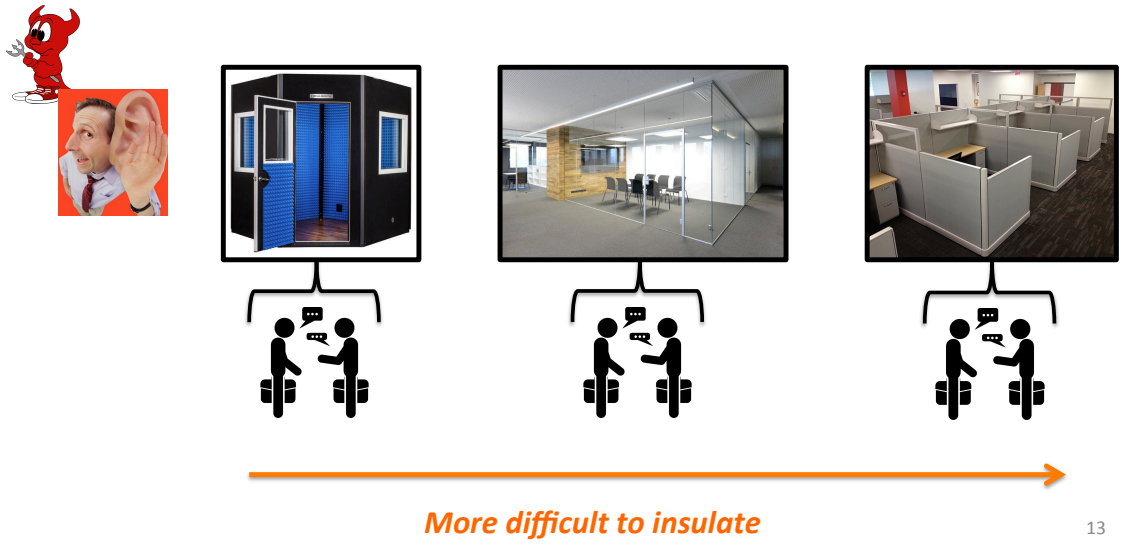


Figure 1.3: Figure depicts an analogy of varying levels physical boundary that insulates conversation to the outside attacker. The physical boundary also governs common secret randomness.

dissimilar hardware across devices. Second, it would also be more desirable to have a *usable* solution by reducing active human involvements in the context-based pairing protocols. Third, I would also want to increase *performance* by reducing the pairing time.

1.4 Constraint on Physical Boundary

In solving the aforementioned challenges of context-based pairing, namely extracting common secret randomness from the Entropy Extraction Module, I need to ensure that the attacker does not perceive the same events as the intended devices. I notice that there is a notion of physical boundary that may separate legitimate devices from the attackers. Hence, the physical boundary insulates certain amount of contextual information that are commonly perceived by intended devices from the attacker outside. For example, Figure 1.3 depicts an analogy by presenting varying levels of physical boundary that insulates conversation that occurs inside the boundary to the outside attacker trying to eavesdrop on the conversation. I note that as the insulation gets weaker (i.e., from soundproof room, to conference room, to an open office setting), it is more difficult to insulate the signals from the attacker. Hence, *physical boundary*, which is an inherent restriction for a given application in its physical world, governs which Singals-of-Opportunity to capture to extract common secret randomness.

Applying this analogy to the IoT and CPS, applications with strict constraint on physical

boundary may be cases where there are *Complete Insulation* of a physical boundary. For example, a Faraday cage may be categorized with such a constraint, where devices inside are completely segregated from the outside world, as no signals are either leaked or injected from and to the cage. Devices inside the boundary can easily authenticate each other knowing that there are only intended devices inside, free from any signal injection from the attackers outside. Following this, less constrained applications may be cases where there are only *Some Insulation* of a physical boundary. For example, a private office or room may be such a constraint, where only intended devices are granted physical access, but wireless signals are also able to penetrate through the walls. Devices in this boundary need to carefully choose the relative context to prove to succeed in authenticating the intended devices. Hence, the devices may prove to each other that they are experiencing similar events that can only be experienced by devices within the boundary. Lastly, applications with more relaxed constraint may be cases with *No Insulation* of physical boundary. For example, a public building such as a shopping mall has almost no constraint, granting access to anyone. Because there are no physical separation, it becomes extremely challenging for the intended devices to authenticate each other.

1.5 Adversary Model

The attacker's goal is to break the main goal of the approach, namely to maliciously forge the verification of physical context during the context-based pairing process. The attacker would try to achieve this goal with the capabilities of capturing relative physical context determined by varying constraint on physical boundary. Specifically, given a constraint, I define a reasonable assumption of the attacker's capability to not be able to capture the relative physical context. For example, devices co-located in *Complete Insulation* such as inside a Faraday Cage are constrained by the physical boundary and so no devices outside can eavesdrop or inject signals inside. Hence, there is a correlation between the spectrum of varying constraint with the varying capabilities of the attacker. There is a high correlation because relaxing the constraint makes verification more difficult as it becomes easier for the attacker to launch attacks. On the other hand, with more constraint, it becomes more difficult for the attacker to forge the relative physical context.

1.6 Contributions of This Thesis

I now highlight my thesis statement and the corresponding research road map. Subsequently, I present a high-level outline of the rest of this thesis.

Thesis statement. *Given a specific constraint on physical boundary, IoT devices identify corresponding Signals-of-Opportunity to extract common secret randomness. In turn, the extracted entropy is used to perform context-based pairing to to bind their identities to their physical relationships, while satisfying scalability, usability, and performance.*

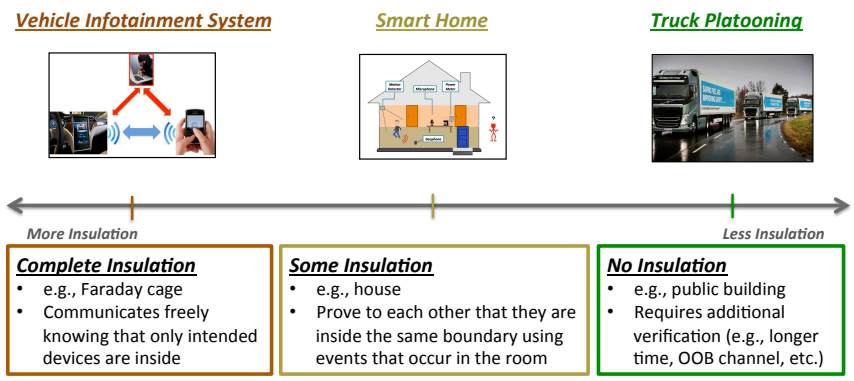


Figure 1.4: I propose to explore different illustrative example scenarios that span through a spectrum of constraint on physical boundary.

In this thesis, I present how I solve the above challenges and achieve the desirable properties by presenting three representative real-world use case scenarios that span across the spectrum of varying constraint on physical boundary. This is depicted in Figure 1.4. Specifically, I demonstrate how to capture the relevant Signals-of-Opportunity to extract common secret randomness in the entropy extraction module for these three scenarios.

1.6.1 Research Road Map

First, I present a pairing scenario between a car's infotainment system and a driver's phone, to illustrate the most constrained solution (of the three examples) – i.e., *Complete Insulation*. Second, I present a smart home scenario to illustrate the less constrained solution – i.e., *Some Insulation*. Finally, I present a truck platooning scenario to illustrate the least constrained solution – i.e., *Least Insulation*.

***Complete Insulation*: Secure Pairing a Smartphone to a Vehicle Infotainment System** – With the increasing popularity of mobile devices, drivers and passengers will

naturally want to connect their devices to their cars. Malicious entities can and likely will try to attack such systems in order to compromise other vehicular components or eavesdrop on privacy-sensitive information. It is imperative, therefore, to address security concerns from the onset of these technologies. While guaranteeing secure wireless vehicle-to-mobile communication is crucial to the successful integration of mobile devices in vehicular environments, usability is of equally critical importance. Several researchers proposed different methods for key agreement between two devices that share no prior secret. However, many of these proposals do not take advantage of the vehicular environment. In Chapter 3, I propose novel approaches to secure vehicle-to-mobile communication tailored for vehicular environments without installing existing hardware for solving this problem. Specifically, I leverage a vehicle's glove compartment as a *complete insulation* analogous to a Faraday cage of a wireless communication. Using this method, I present novel security protocols and their security analysis and provide implementation and user study results demonstrating the feasibility and the usability of our solution.

***Some Insulation*: Autonomous Secure Pairing of Smart Home Devices** – Previous work on context-based pairing solutions increase the usability of IoT device pairing by eliminating any human involvement in the pairing process. This is possible by utilizing on-board sensors (with same sensing modalities) to capture a common physical context (e.g., ambient sound via each device's microphone). However, in a smart home scenario, it is impractical to assume that all devices will share a common sensing modality. For example, a motion detector is only equipped with an infrared sensor while Amazon Echo only has microphones. In Chapter 4, I present our novel context-based pairing mechanism that uses time as the common factor across differing sensor types. By focusing on the event timing, rather than the specific event sensor data, our solution creates event fingerprints that can be matched across a variety of IoT devices. I propose our solution based on the idea that devices co-located within a physically secure boundary (e.g., single family house) can observe more events in common over time, as opposed to devices outside. Devices make use of the observed contextual information to provide entropy for our pairing protocol. In this chapter, I present our design and implementation details and evaluate its effectiveness as an autonomous secure pairing solution. Our implementation demonstrates the ability to sufficiently distinguish between legitimate devices (placed within the boundary with only *some insulation*) and attacker devices (placed outside) by imposing a threshold on fingerprint similarity.

***No Insulation*: Verifying Relative Vehicle Positioning** – Truck platooning is emerging as a promising solution with many economic incentives. However, securely admitting

a new vehicle into a platoon is an extremely important yet difficult task. There is no adequate method today for verifying physical arrangements of vehicles within a platoon formation. Specifically, I address the problem of a *platoon ghost attack* wherein an attacker spoofs presence within a platoon to gain admission and subsequently execute malicious attacks. To address such concerns, I present in Chapter 5 a novel autonomous platoon admission scheme which binds the vehicles' digital certificates to their physical context (i.e., locality). Our solution exploits the findings that vehicles traveling together experience similar context to prove to each other over time that they are co-present. Specifically, they experience similar road (e.g., bumps and cracks) and traffic (e.g., acceleration and steering) conditions as opposed to vehicles traveling even in adjacent lanes. Hence, the vehicles have the *least insulation* among the three projects. Specifically, our approach is based on the ability for vehicles to capture this context, generate fingerprints to establish shared keys, and later bind these symmetric keys to their public keys. I design and implement our protocol and evaluate it with real-world driving data. Our implementation demonstrates that vehicles traveling in adjacent lanes can be sufficiently distinguished by their context and this can be utilized to thwart platoon ghost attacks and similar misbehavior.

Risks of Sensing for Cyber-Physical Systems – In addition to leveraging sensing information to help bind identities to the physical relationships, I studied how sensing information can pose potential security threats from several research projects during my Ph.D. program. In this thesis, I present one of the additional work in this theme. To exemplify this line of work, I present one project that highlights how an attacker may invade personal privacy in a smart home using a fusion of available sensor data. Specifically, in Chapter 6, I propose a new side-channel attack, where a network of distributed non-acoustic sensors can be exploited by an attacker to launch an eavesdropping attack by reconstructing intelligible speech signals. In this chapter, I demonstrate the feasibility of speech reconstruction from non-acoustic sensor data collected offline across networked devices. Unlike speech reconstruction which requires a high sampling frequency (e.g., > 5 KHz), typical applications using non-acoustic sensors do not rely on richly sampled data, presenting a challenge to the speech reconstruction attack. Hence, our solution leverages a distributed form of Time Interleaved Analog-Digital-Conversion (TI-ADC) to approximate a high sampling frequency, while maintaining low per-node sampling frequency. I demonstrate how distributed TI-ADC can be used to achieve intelligibility by processing an interleaved signal composed of different sensors across networked devices. I present our implementation details and evaluate reconstructed speech signal intelligibility via user studies. The word recognition accuracy is as high as 79%. Though some additional

work is required to improve accuracy, our results suggest that eavesdropping using a fusion of non-acoustic sensors is a real and practical threat.

1.6.2 Thesis Outline

The remainder of this thesis is organized as follows. I present the related work with respect to the varying physical boundary constraint in Chapter 2. Subsequently, I present the three research projects from most to least constrained physical boundaries in Chapters 3, 4, and 5. In Chapter 6, I present one of the security vulnerabilities due to sensing. In Chapter 7, I summarize my contributions, and present future directions of my research.

Chapter 2

Related Work

In this chapter, we investigate three related work that span the spectrum of varying constraints on physical boundary and highlight their limitations in achieving the main challenges.

As the most constrained scenario among the three related work, we present Message-In-a-Bottle (MIB) [84], which leverages a Faraday cage, a special hardware device that ensures authenticity and secrecy of the communication inside the cage. The cage prevents any wireless signals communicating between the sensor nodes inside to be leaked to attackers outside the cage. Also, attackers are infeasible to inject wireless signals into the cage. The main drawback of this approach is that it requires specialized hardware such as the Faraday cage.

Second, as a relatively relaxed constraint scenario, we present Distance Bounding (DB) [39], which leverages estimated distance via wireless signals to be used as a "virtual" boundary. Although there are no physical boundary like MiB, DB requires that a human verifies if there are non-intended devices within the virtual boundary. Similarly, we also present HAPADEP [125], which leverages audio channel as an Out-of-Band Channel to perform a secure pairing of devices. It first exchanges the public keys of two smartphones via sound, using built-in speakers and microphones. In the verification phase, the two devices reveal data commitments by emitting another sequence of sound signals. The user is required to verify if the two sound signals are identical. Similar to DB, there are no physical boundary. However, it also requires that a human verifies if there are non-intended devices within the virtual boundary.

Third, as a most relaxed constraint scenario, we present Zero Interaction Authentication (ZIA) [99], which proposes to perform secure pairing across devices in a smart home using either a pair of microphone or light sensors. The core idea is that the devices inside the house would share similar context as opposed to devices in a neighbor's house. Similarly,

Ambient Audio [117] presents a similar idea by utilizing short audio signals as contextual information shared across a pair of microphones and leverages fuzzy commitment schemes to perform a secure pairing. However, both of these papers make strong assumptions that the devices inside are all equipped with the same sensor type. Hence, this assumption limits the practicality and scalability especially in a smart home environment, where the number of devices are expected to increase to hundreds in the next few years. In this thesis, we propose to also achieve such scalability by reducing the assumptions of hardware used.

Chapter 3

Most Constrained: Vehicle Infotainment System

We identify the pairing scenario as the most constrained one because the methodology used in this chapter entails use of a car and its glove compartment as a tightly managed physical boundary.

3.1 Problem Definition

With the proliferation of wireless devices using Wi-Fi and Bluetooth technologies, security of their communication is a vital concern as numerous real-world attacks have been reported [116]. Insecure wireless communication may allow attackers to eavesdrop or launch Man-in-the-Middle (MitM) attacks, impersonating legitimate communicating devices.

Efforts to eradicate such attacks have inspired many research proposals as well as industrial solutions, namely to provide secure pairing between devices by "bonding" them to establish a secure channel. However, it is still difficult for human users to easily determine which devices are being paired because of the invisible nature of wireless communication. Hence, researchers propose demonstrative identification, which affirms to the human user which devices are actually communicating leveraging out-of-band channels. [30].

However, many naive solutions attempting to establish such secure pairing for any two devices introduces a trade off. In many cases, increasing security leads to decreased usability, which becomes a significant hindrance for wide adoption of the technology by the general public. On the other hand, decreased usability may cause a security breach in these protocols. This is exemplified by the use case scenario when a user tries to pair her phone with a

friend's phone using Bluetooth. The state-of-the-art solutions require the user to either copy a passkey displayed on one device to the other, compare two passkeys displayed on both devices, or to enter a hard-to-guess passkey on both devices. However, the security of such protocols often rely on the passkey not being repeated or easy-to-guess, requiring the users to input hard-to-guess passkeys to guarantee the protocol security [85]. These designs, however, lead to multiple problems in practice. Many devices actually display a repeated and/or easy-to-guess passkeys (e.g., 000000, 123456, etc.) [136]. Also, many users tend to make fatal mistake of inputting easy-to-guess passkeys [132].

In this section, we delve into a specific problem of vehicular environments. The proliferation of smartphones coupled with emerging smarter vehicles allows constant exchange of sensitive information over wireless communication. For example, different automotive manufacturers and smartphone companies established Car Connectivity Consortium (CCC) and have formed *Mirror Link*, a standard for integrating smartphones and the vehicles to enable access to the phones using car's control, display, and speakers [41]. In addition to pairing with personal cars, we expect more frequent pairing use cases for widely deployed rental car services – both traditional and short-term rental cars (e.g., Zipcar).

Unfortunately, coupling of smartphones and vehicles introduces a new avenue of potential attacks if the wireless channel is not secured. Although launching such attacks may not seem plausible at a first glance, they are certainly within the realm of possibility especially for high-value targets (e.g., celebrities, politicians, etc.) that provide more incentives for the attackers. Furthermore, such targets are more likely to drive luxury vehicles that embrace next-generation vehicle-to-mobile convergence systems. Are current cars effectively protected from remote attackers attempting to compromise vehicular components? Can we be convinced that the sensitive information in our vehicles is not being maliciously transmitted to attackers in other nearby cars on the road, or in parking lots connected via Bluetooth or Wi-Fi?

To address these problems, we present *MVSec*, the first secure key agreement scheme tailored specifically for vehicular environments, providing strong security guarantees and easy usability. *MVSec* leverages out-of-band channels such as sound or light because commodity hardware such as LED, ambient light sensor, speaker, and microphone are readily available in cars and/or smartphones. *MVSec* allows a user, typically the driver, to simply press a button on each device (the car and the phone) to initiate the protocols. For the protocols that leverage sound, all the user needs to do is to simply verify that both the car and the intended mobile device emit a short beep. Similarly, for protocols that leverage light emission, the user simply needs to place the mobile device in the glove compartment for a short amount

of time.

3.2 System Models

This section presents the goals we plan to achieve given the constraints, lists the assumptions we hold, and discusses the attacker model. The main goal of *MVSec* is to present a complete system that provides a secure and usable communication between the car and the smartphone. If an attacker is present and launches an attack, it will be clear to the user that an error has occurred, so that the user can immediately abort the pairing process. The main properties *MVSec* tries to achieve are the following. *MVSec* achieves **secrecy** by allowing the driver's phone and the car to hide information from unintended devices. It also achieves **authenticity** and **integrity** by allowing the driver's phone and the car to validate that unaltered data arrived from the claimed sender. *MVSec* also achieves **demonstrative identification** by enabling the user to explicitly be aware of which devices are actually communicating via the wireless communication.

We also categorize some of the constraints that pose challenges in achieving the aforementioned goals. The phone and the car initially do not share any prior secret, nor depend on any Trusted Third Party (TTP) for exchanging the secret. In addition, *MVSec* incurs minimal hardware cost by leveraging available hardware commonly installed in today's cars and smartphones to communicate via out-of-band (OOB) channels (discussed further in Section 3.3).

3.2.1 Adversary Model

This subsection presents the attacker model by describing the attacker goals and capabilities.

Attacker Goals The goals of the attacker is to break the security properties that *MVSec* aims to achieve, namely to break secrecy and authenticity of vehicle-to-mobile communication.

Attacker Capabilities We assume that the attacker can perform both passive and active attacks. A passive attacker can perform attacks without actively participating in the protocol, such as eavesdropping. An active attacker follows a Dolev-Yao attacker model who are able to perform various types of attacks in addition to eavesdropping – data injection attacks, denial-of-service, man-in-the-middle (MitM), etc. In this thesis, we concentrate on defending against the MitM attack.

There are two types of attackers that we envision in a secure pairing environment with

vehicle and mobile devices. Different attackers are limited by different capabilities given corresponding constraints.

(1) Attacker inside the vehicle. In this case, an attacker is present inside the vehicle. This type of attacker can launch powerful attacks because the attacker has access to the visual display in the vehicle. Computational power is not a limitation for this type of attacker because he can have remote access to powerful computational devices. The only limitation would be the bandwidth of the network connectivity.

(2) Attacker is outside of the vehicle but within wireless transmission range (<100 meters away). Attackers who are outside of the car can still perform powerful attacks. This type of attacker is constrained in many aspects compared to the first type as the attacker cannot see the visual display and listen to sound emissions from the vehicle and the mobile devices, and cannot launch out-of-band channel attacks. The attacker can still perform considerable attacks such as jamming, eavesdropping, and man-in-the-middle attacks, especially if equipped with powerful devices such as a high-gain antenna.

3.2.2 Assumptions

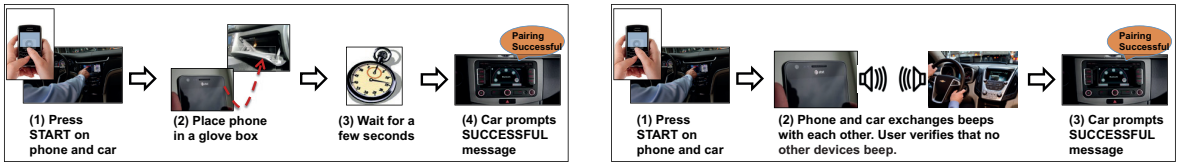
Given the specific vehicular setting, we make the following assumptions to achieve the aforementioned goals. We assume that the OOB channel *does not require user diligence*. This is a necessary assumption to ensure high usability. We also assume that there is *no malware* on the vehicle or mobile device. If there is malicious code on the mobile device, a secure pairing protocol cannot establish a secret because attackers can gain the shared secret through the malware.

3.3 Capturing Contextual Cues

This section presents the overview of the *MVSec* protocols, discusses the OOB channel selection, and then delves into the protocol details. The main goal of *MVSec* is to allow a user to securely pair his/her smartphone with a vehicle such that an attacker will not successfully launch MitM attacks.

To achieve this goal, we first need to overcome the challenge of providing *demonstrative identification*, to ensure that the vehicle and the intended smartphone are in fact communicating with each other. We leverage out-of-band (OOB) communication channels as a solution. Different from the in-band channels used by the devices, e.g., Wi-Fi or Bluetooth,

an OOB channel is a separate communication medium between the communicating devices (e.g, humans, light, sound, vibrations).



(a) Strong OOB Channel (light)

(b) Weak OOB Channel (sound)

Figure 3.1: Overview of *MVSec* using light and sound as OOB channels

3.3.1 Out-of-band Channel Selection

MVSec leverages two types of OOB channels for different protocols. According their characteristics, they are categorized into *strong* and *weak* OOB channels.

Strong OOB Channel. A strong OOB channel guarantees both *secrecy* and *authenticity*. We select **light** in a vehicle's closed glove compartment as the strong OOB channel because it helps to provide both of these security properties with minimal human involvement. We assume that the glove compartment does not leak any light signal, thus provides secrecy. This channel also provides authenticity because only the vehicle will emit light signals. This is because no other device is inside the compartment as the driver first verifies that other devices are not placed inside the compartment during protocol execution.

In addition to considering the security properties, we choose light because of the following requirements of the OOB channel. The OOB channel needs to (1) be readily available in vehicles and smartphones today in order to be easily deployable, and (2) provide high usability, i.e., the OOB channel needs to have a relatively fast data rate and be easy and pleasant to use (i.e., should not require user diligence nor annoy the users). We define relatively fast data rate to be faster than the OOB channel used as baseline case, which is manual human input. This OOB channel allows such usability because the only task that the user performs is to press a button on both the vehicle and the smartphone, and place the smartphone inside the glove compartment. After waiting for a few seconds, during which the vehicle transmits signals via blinking lights to the smartphone, the pairing process successfully completes. The steps are depicted in Figure 3.1(a).

Weak OOB Channel. A weak OOB channel provides only *authenticity*. We select **sound**

signals as the weak OOB channel. This channel helps to provide authenticity, again, with minimal human involvement, because we assume that it is easy for the user to identify that the sound beeps are originating only from the intended devices (e.g., vehicle and driver's smartphone). If an unintended device beeps, the user simply aborts the protocol. We assume that the beeps are sufficiently long and loud enough for the user to easily identify the origin of the beeps. We assert that this is a realistic assumption, because smartphone users generally distinguish who's phone is ringing when (s)he hears a phone ring. We also use sound signals because of the ubiquitous deployments of microphones and speakers in vehicles and smartphones. Figure 3.1(b) depicts the steps the user performs. We provide different variation of protocols using different cryptographic primitives that leverage the two aforementioned OOB channels.

3.4 Design and Implementation

This section presents the design and implementation details of *MVSec*.

3.4.1 *MVSec* Protocols

This subsection describes three *MVSec* protocols that leverage light and sound signal in a closed compartment as the OOB channel. We present the underlying cryptographic primitives of these protocols and compare them to select the most efficient protocol. Table 3.1 denotes the notations used in the *MVSec* protocols.

3.4.1.1 Strong OOB Channel

We now describe protocol details leveraging the light as an OOB channel. This protocol makes use of the Encrypted Key Exchange (EKE) [32] and is depicted in Figure 3.2.

***MVSec*-I: Protocol leveraging Encrypted Key Exchange (EKE).** A conventional EKE scheme allows two participating entities to use a shared low-entropy password to derive a temporary shared key that can be used to authenticate the key exchange messages. We use a variant of the EKE scheme by treating a short shared secret K_s (20 bits) as a low entropy password. K_s is first transmitted via the light signal in Step 2. In Steps 3 and 4, both the vehicle and the smartphone transmit their DH public keys encrypted with K_s. Then the vehicle and the mobile device also performs key confirmation in Steps 4 and 5.

Notation	Description
$\overset{Light}{\Longrightarrow}$	Strong OOB channel using light in a closed glove compartment
$\overset{Sound}{\Longrightarrow}$	Weak OOB channel using sound
$\overset{BT}{\longrightarrow}$	In-band Bluetooth channel
$M_K(x)$	MAC (e.g., HMAC) computed over the input x, using key K
g^x	Diffie-Hellman public parameter (omitted *mod p* for clarity)
$H(x)$	Cryptographic hash (e.g., SHA-3) of input x
$\{x\}_K$	Symmetric encryption (e.g., AES) of input x using key K
$[x]_i$	Truncated i bit string of input x
$\{0,1\}^i$	Random binary string with length i

Table 3.1: Notations for MVSec protocols

3.4.1.2 Weak OOB Channel

MVSec leverages *Short Authenticated Strings (SAS)* [88, 109, 135] which uses commitment/decommitment schemes prior to transmitting the short hash comparisons for verification. This approach is preferred over a naive approach of sending short hash values over the weak OOB channel for verification. The reason is that attackers may be able to launch attacks to find hash collisions.

***MVSec*-II: SAS with random nonce.** Figure 3.3 presents first of the two protocols that leverages weak OOB channel. This protocol uses short authenticated strings (SAS) with random nonces.

In Steps 2 and 3, the vehicle and the mobile device generate and exchange commitments. The commitments are the hashes of each party's DH public key concatenated with its random nonce, n_A and n_B. Steps 4 and 5 allows the vehicle and the mobile device to transmit the decommitment in order to verify public keys with stored commitments. Once both parties verify the hash by comparisons, they will transmit the SAS messages mutually through the weak OOB channel in Steps 6 and 7. Note that the vehicle can also use light in a glove compartment to transmit this SAS message, although the secrecy is not needed in this protocol. Meanwhile, the vehicle and the smartphone both verify the received SAS value is equal to the SAS value they sent out. If two SAS values match, both devices generate their

***MVSec*-I: Protocol using EKE**

1. $User$	: Presses start buttons on A and B. Places B in the glove compartment.
2. A	: $K_s \xleftarrow{R} \{1,0\}^\ell$
$A \overset{Light}{\Longrightarrow} B$	: K_s
3. $A \xrightarrow{BT} B$	: $\{g^a\}_{K_s}$
B	: Decrypts $\{g^a\}_{K_s}$ to get g^a; Computes shared key $K' = (g^a)^b$.
4. $B \xrightarrow{BT} A$	: $\{g^b\}_{K_s} \| M_{K'}(n_A)$ where $n_A = H(\{g^a\}_{K_s})$
A	: Decrypts $\{g^b\}_{K_s}$ to get g^b; Computes shared key $K = (g^b)^a$; $M_{K'}(n_A) \overset{?}{=} M_K(H(\{g^a\}_{K_s}))$; aborts if verification failed.
5. $A \xrightarrow{BT} B$	: $M_{K'}(n_B)$ where $n_B = H(\{g^b\}_{K_s})$
B	: $M_{K'}(n_B) \overset{?}{=} M_K(H(\{g^b\}_{K_s}))$; aborts if verification failed.

Figure 3.2: Key agreement protocol between a vehicle (A) and a smartphone (B) using EKE with light as the strong OOB channel. In practice, $\ell = 20$.

individual DH ephemeral key, K and K'. Such as the previous protocol, the key confirmation procedure would perform in Steps 8 – 11.

3.4.2 Implementation

We demonstrate working *MVSec* protocols using the Android platform. We use two Motorola Droid 1 phones running Android 2.2.3 (Froyo) - one to simulate the car and the other to represent the driver's smartphone respectively.

3.4.2.1 *MVSec* Pairing Walk-Through

We now provide a walk-through of the *MVSec* pairing protocols by describing our implementation prototypes leveraging both the weak and strong OOB channels (described in Section 3.4.1). Figure 3.4 presents chronological protocol steps for the protocols that leverage sound which is a weak OOB channel. First, both devices prompt the user with instructions and a "Pair Now" button on the car. Once the user presses this button, the two devices will first initiate pairing messages over Bluetooth, as shown in *MVSec*-II protocol. Then the two devices transmit each other's SAS messages over the authentic OOB channel, which is via the audio channel. As soon as the car finishes emitting the beep, the smartphone starts beeping, and the car listens for the beep. In Section 3.4.2.2, we present a detailed description of the encoding and decoding of the sound pulses. After the SAS messages have been exchanged, the two devices complete the protocol by exchanging the key confirmation messages again

***MVSec*-II: Protocol using SAS with Nonce**

1. $User$: Presses start buttons on A and B. Aborts if devices other than A or B beep during execution.
2. A : Nonce $n_A \xleftarrow{R} \{1,0\}^{\ell}$.
   $A \xrightarrow{BT} B$: $C_A = H(g^a || n_A)$.
3. B : Nonce $n_B \xleftarrow{R} \{1,0\}^{\ell}$.
   $B \xrightarrow{BT} A$: $C_B = H(g^b || n_B)$.
4. $A \xrightarrow{BT} B$: $g^a || n_A$.
   B : $C_A \stackrel{?}{=} H(g^a || n_A)$; aborts if verification fails.
5. $B \xrightarrow{BT} A$: $g^b || n_B$.
   A : $C_B \stackrel{?}{=} H(g^b || n_B)$; aborts if verification fails.
6. $A \overset{Sound}{\Longrightarrow} B$: $SAS_A = n_A \oplus n_B$.
   B : $SAS_B = n_B \oplus n_A$
   $SAS_A \stackrel{?}{=} SAS_B$; aborts if verification fails.
   Computes shared key $K = (g^a)^b$;
7. $B \overset{Sound}{\Longrightarrow} A$: SAS_B.
   A : $SAS_B \stackrel{?}{=} SAS_A$; aborts if verification fails.
   Computes shared key $K' = (g^b)^a$.

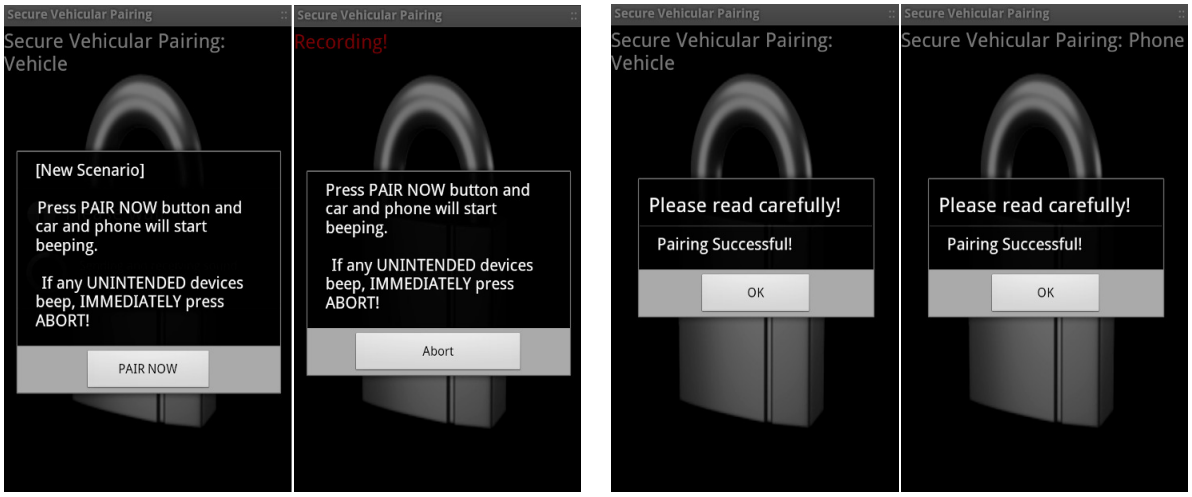
Key confirmation (check $K' \stackrel{?}{=} K$)

8. A : Nonce $n'_A \xleftarrow{R} \{1,0\}^{\eta}$.
   $A \xrightarrow{BT} B$: $n'_A || M_{K'}(n'_A)$
9. B : Nonce $n'_B \xleftarrow{R} \{1,0\}^{\eta}$.
   $B \xrightarrow{BT} A$: $n'_B || M_K(n'_A || n'_B)$
10. A : $M_K(n'_A || n'_B) \stackrel{?}{=} M_{K'}(n'_A || n'_B)$;
    abort if confirmation fails.
    $A \xrightarrow{BT} B$: $M_{K'}(n'_B)$
11. B : $M_{K'}(n'_B) \stackrel{?}{=} M_K(n'_B)$;
    abort if confirmation fails.

Figure 3.3: Key agreement protocols between a vehicle (A) and a smartphone (B) using SAS with random nonce, leveraging weak OOB channel providing only authenticity. In practice, $\ell = 20$ and $\eta = 256$ (HMAC-SHA3).

over the in-band Bluetooth channel. After the entire protocol has successfully completed, the two devices prompt the user of a pairing successful message.

Figure 3.5 provides a similar chronological protocol steps, but leverages light in a glove compartment which is a strong OOB channel. Once the user initiates the pairing process, the device simulating the vehicle (car for short) will start transmitting light pulse by varying the light intensity levels of the screen. Section 3.4.2.3 also provides implementation detail of the light channel. The smartphone will capture the varying light intensity levels in this step. Then the car and the phone will now exchange messages over Bluetooth, to complete *MVSec*-I protocol. Once the protocol completes, both devices prompt the user of a successful pairing message.



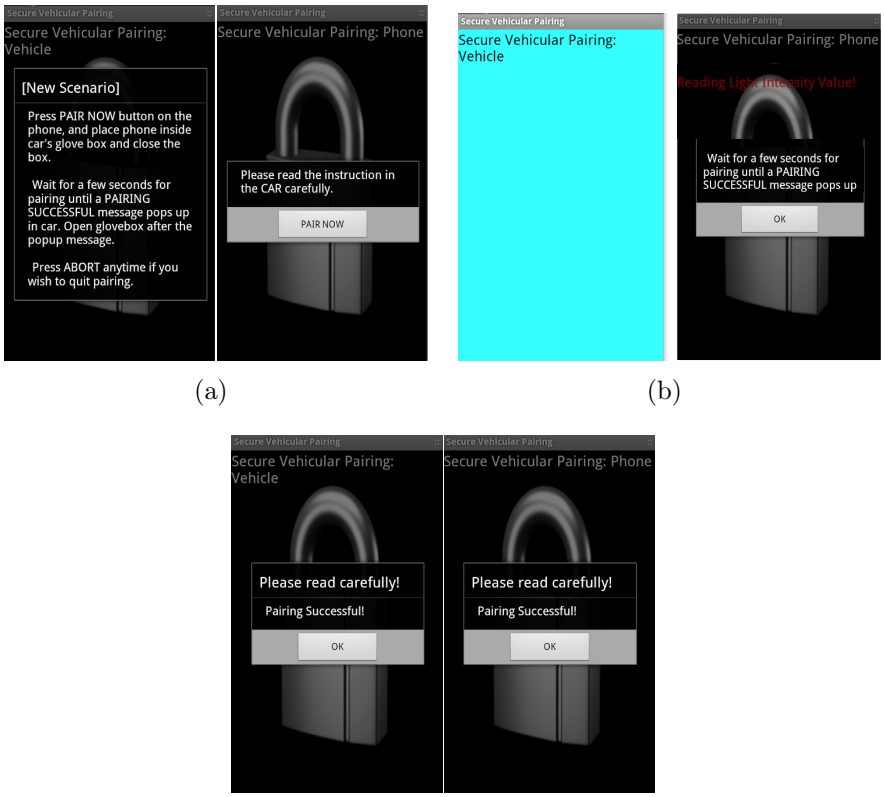
(a) (b)

Figure 3.4: Execution flow of *MVSec* protocol leveraging a weak OOB channel (via sound). (a) Upon pressing a start button, the phone simulating the vehicle (car for short) displays instructions to initiate pairing and to abort if any unintended devices beep. (b) User is prompted with pairing successful message.

3.4.2.2 Audio Channel

MVSec leverages sound as an weak OOB channel for the following reasons. First, as described in Section 3.4.1.2, the SAS messages only need to be authenticated, but not require to be secret. The audio channel provides authenticity because the driver can easily determine the source device of the sound beeps inside the car - i.e., whether the beep is originating from the car speakers and his intended mobile device, as opposed to other unintended devices (e.g., passenger smartphone). Second, the necessary hardware are already available in the car and the phone. This is because all cars and phones have speakers and microphones.

In order to transmit 20 bits of the SAS message, we first encode the data into eight different frequencies, allowing 20 bits to be encoded to 8 pulses. It takes roughly 800 ms to transmit a pulse (including the pause), so it takes roughly 5.6 seconds to transmit all 20 bits of data. For example, when the transmitter transmits 0x93759 as the SAS message, it is first encoded the message to '2233531' in base 8. On the receiver's side, we leverage Android's AudioRecord class to record the sound signal. Once the signal is recorded, we filter the signal by applying Goertzel algorithm [40, 60] for the eight target frequencies. We use eight frequencies evenly distributed from 900Hz to 1600Hz. The aggregate of the filtered frequencies represented by their magnitude squared (mag^2), is depicted in Figure 3.6. Each



(a) (b)

(c)

Figure 3.5: Execution flow of *MVSec* protocol leveraging a strong OOB channel (via light in a glove compartment). (a) Upon pressing a start button, the phone simulating the vehicle (car for short) prompts the user instruction to place the phone in the glove compartment. (b) The car is emitting light signal inside the glove compartment and the phone is detecting the signal. (c) User is prompted with pairing successful message.

spike represents the pulse that correlates to a base 8 number. To finalize the decoding phase, we process the pulses by apply a sliding window technique to the mag^2 values. The sliding window algorithm is triggered when the mag^2 value exceeds a certain threshold, th. Upon triggering the sliding window algorithm, we check to see if the mag^2 value exceeds th within a certain window size, wnd. If the value exceeds th, we increment a counter until it exceeds the detection threshold, dth. We then classify this window as a legitimate sound pulse. Using empirical analysis, we set wnd=4000, th = 40, and dth=200. The described processing increases the detection accuracy, and reduce false positives, and successfully decodes the pulses to the correct '2233531'(0x93759).

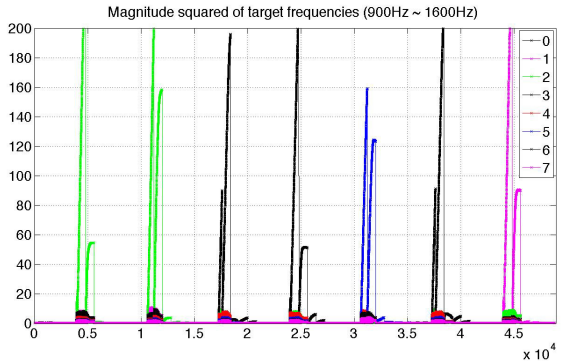


Figure 3.6: Magnitude squared of target frequencies 900Hz - 1600Hz. The decoding algorithm will process this to '2233531' (== 0x93759).

3.4.2.3 Visual Channel

MVSec leverages a strong OOB channel to transmit a short, temporary secret key to defend against the MitM attack. This OOB channel leverages the light bulb in a closed glove box to transmit messages, which will be detected by an ambient light sensor on the driver's smartphone. An Android smartphone is equipped with an ambient light sensor that measures the light intensity experienced by the phone. This sensor is generally used to detect light intensity for automatic brightness control and screen locking. We leverage Android's SensorManager class to implement the prototype using the ambient light sensor. In our implementation, we fully implement the driver's smartphone, and simulate the car's glove box light source, by using another Android device, by varying the light intensity of the screen.

When the driver presses the start button on each device to initiate the protocol, the car will emit a sequence of light signals to the driver's smartphone. The signal is an encoding of a short temporary key (20 bits) as described in the protocol details in Section 3.4.1. Accounting for the low resolution of the ambient light sensor on the smartphones, the current prototype leverages four intensity levels to encode the corresponding bits: low, medium, high, and pause. Each level corresponds to the following lux values received by the receiver's ambient light sensor – 10 lx, 40 lx, 90 lx, and 160 lx. Due to the low sampling rate of the ambient sensor in the driver's phone, the car transmits one intensity level for every two seconds (one second for intensity value and another second for the pause bit), and takes a total of 26 seconds to transmit (20 bits encodes to 13 pulses). However, we envision that more responsive ambient sensors installed in newer phones will increase the speed.

MVSec uses the ambient light sensor as a proof-of-concept. However, we envision that using other sensors to read the light signal (e.g., camera) would increase the overall detection

time and improve the performance.

3.5 Evaluation

This section provides the evaluation of the usability, as well as the OOB channel detection accuracy. We present the results of the user study conducted by describing the participant profile, study process, and analysis of the results. We also evaluate how accurate the OOB channel is in terms of the detection accuracy.

3.5.1 Usability Analysis

The main goal of this user study is to determine the usability of the MVSec. Specifically, we design our study to verify (1) whether MVSec reduces user errors as well as pairing timing, and (2) the user's perception of MVSec being more secure and simple to use compared to other solutions.

3.5.1.1 Demographics

We recruited 23 participants from different sources such as Craigslist and a university mailing lists. The participant pool varied in gender, age, and education background. The participants' age range was 20–59; 13 are in twenties, 6 are in thirties, and 4 are in more than forties. These participants include 12 male and 11 female. Among 23 participants, 10 have undergraduate degree (e.g., master or doctorate degrees), 13 have college degree, and one participant has only high school diploma.

3.5.1.2 User Study Process

Participants are invited to the driver's seat in a car to perform user study. We present to them with two phones – one to simulate the car's control unit (P_{car}) and the other to be used as the driver's smartphone (P_{driver}). P_{car} is attached to the car's dashboard to simulate the vehicle's infotainment system. We designed both the *light (L)* and *sound (S)* MVSec scenarios to be tested for the user study. Although we fully implemented the working prototype for L scenario, we simulated the L scenario for the user study by asking the user to place P_{driver} into the glove compartment and explained to them that the light in the compartment will be emitting secret light signal to P_{driver}.

Scenarios	Description
Light (L)	Light in a glove box. MVS-I is represented by this scenario. Participant are informed that the light in the glove compartment will transmit secret message to the smartphone's ambient light sensor.
Sound (S)	Bi-directional OOB channel using audio signals. MVS-II is represented by this scenario.
Sound Attack (SA)	An unintended device beeps while presenting the *Sound* scenario, and the user is expected to abort after noticing the beep.
Choose-and-Enter (CE)	Base line case that asks the user to create a passkey and enter it on both devices.
Compare-and-Confirm (CC)	Base line case that asks the user to compare the displayed numbers on both devices.
Compare-and-Confirm Attack (CCA)	Different numbers are displayed in the *Compare-and-Confirm*, and the user is expected to notice the difference.
Copy (C)	Base line case that asks the user to copy a displayed number from one device to the other.

Table 3.2: Different scenarios presented to the participants.

For comparison, we implemented three baseline cases that are currently used as Bluetooth pairing schemes in vehicles. The three cases are *choose-and-enter (CE)*, *compare-and-confirm (CC)*, and *copy (C)*. *CE* allows the user to choose a hard-to-guess number and enter it on both of the devices. *CC* allows the user to compare the numbers displayed on each of the devices. *C* allows the user to copy a displayed number on the car, and input it into his phone.

In addition to these five scenarios, we also added two attack scenarios – one for MVSec and the other for the baseline case. First, we present an attack on *sound* by having an unintended device beep, when the participant is performing sound pairing scenario, and test if the participant is able to detect the beep from the unintended device and aborts the pairing process. Second, we present an attack on *CC* by presenting two numbers that are different by a digit, and test if the participant can determine the difference. To reduce bias between the subjects, we present the seven scenarios in random order for different participants. Table 3.2 lists all seven scenarios presented to the participants.

3.5.1.3 Study Results

During the execution of the scenarios, we measure the following two outputs for comparison – *error rate* and *time*. For non-attack scenarios (i.e., *L*, *S*, *CE*, *CC*, and *C*), we claim that

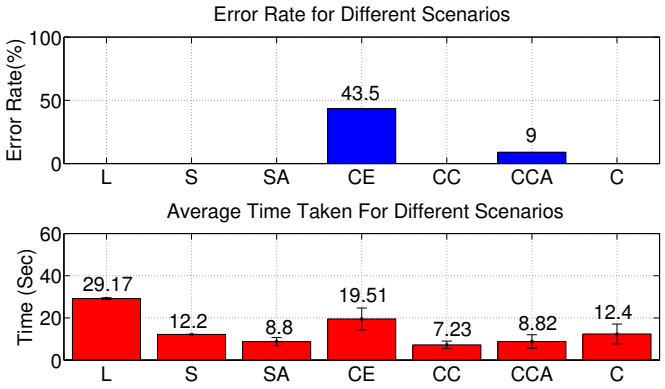


Figure 3.7: Error rate and time measurements of different study types. Note attack scenarios for both S and CC are included.

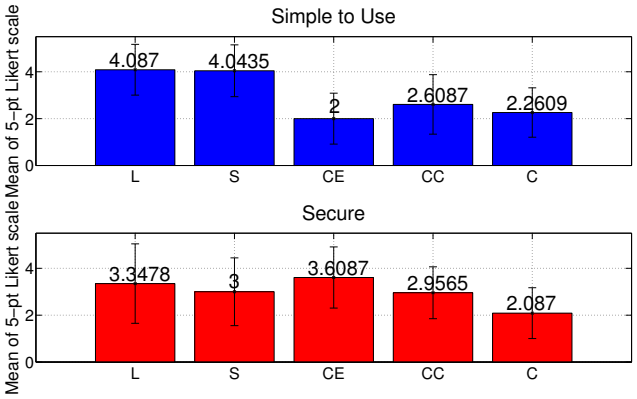


Figure 3.8: Post-test questionnaire results that rate user's perceptions for simplicity/security.

an error occurs when the participant performs tasks in an incorrect manner resulting in an unsuccessful pairing. For attack scenarios for S and CC, an error occurs if the participant does not detect a problem, and continues the pairing procedure without aborting. Figure 3.7 depicts the comparison of the six scenarios with respect to error rate and timing.

The first graph in Figure 3.7 illustrates that the error rate is around 45% for CE, which is a significant percentage. This is because many participants chose easy-to-guess six digit number, when asked to come up with a six digit passkey. Because the security of this approach depends on the passkey to be unpredictable, this demonstrates a clear security problem. We also observe that for the attack scenario of *compare-and-confirm* (Scenario SA), about 10% of the participants mistakenly accepted different values displayed on the devices to be the same. However, we did not find any error caused by the participants when pairing via the

Average background noise	Success rate	Human perception of background noise
65 dB	90%	Quiet inside car, with some noise from garage ventilation
75 dB	92%	Music volume is loud, but beeps from phones audible
85 dB	86%	Music volume is very loud, beeps from phones are not audible at times
95 dB	2%	Music volume is extremely loud, beeps from phones are not audible at all

Table 3.3: Comparison of sound signal detection accuracy, when varying the background noise level by performing 50 trials per dB.

L and S. More interestingly, during the attack scenario of S, all participants distinguished the beeps from the intended devices as opposed to the unintended device, and pressed abort button as instructed.

The lower graph in Figure 3.7 depicts the average time taken for different scenarios. On average, L took around 29 seconds, which is the longest to complete, due to the low resolution of the ambient light sensor. CE followed L with around 20 seconds of average completion time. This is because the participants had to come up with a six digit passkey, and enter the number twice, once on each device. C and S took about the same time of around 12 seconds. The fastest average completion time was CC, because this scenario did not require the user to enter any numbers on the devices.

Upon completion of all seven scenarios, we asked the participants to rate the scenarios (excluding the attack scenarios) with a five point Likert scale for *simplicity* and *security* (scale from 1 to 5: 1 being the least simple/secure unto 5 being the simplest/most secure). Figure 3.8 depicts the average of five point Likert scale. It is interesting to note that both L and S have significantly higher average (both above average value 4) than the baseline cases for simplicity, despite the fact that L took the longest time to complete. It is also interesting to note that the user perception for security are relatively well distributed among different scenarios, fortifying the fact that the participants well represent average users without security expertise.

With the aforementioned results, we claim that MVSec protocols provide a clear usability advantage over the baseline cases, which are used as industry standards in many of the vehicle-to-mobile pairing schemes. We find that MVSec simplifies user experience, while significantly reducing error rate.

3.5.2 OOB Detection Accuracy

We now present the detection accuracy of the audio channel of our prototype implementation discussed in Section 3.4.2.2. The detection accuracy of audio signals, even in the presence of background noise, are important because drivers will be faced with similar situations in real-world use cases.

We conducted the test inside a vehicle, and varied the background noise level, by changing the volume while repeating the same song segments. We measured the average background noise by recording the segment via Skypaw's Multi Measures – Decibel smartphone application [122]. We transmitted a 20 bit number via the audio channel between two smartphones separated by 11 inches apart. We repeat the transmission for 50 trials per each background noise level. The test results are summarized in Table 3.3. As shown in the table, the detection rate is significantly high even when exposed to a relatively loud background noise. However, we find that the detection accuracy drops significantly when the noise level reaches a threshold where the beeps from the phones are almost inaudible to human ears. From this experiment, we claim that the sound detection of MVSec prototypes are robust to be used in practice.

3.6 Related Work

Many researchers have investigated the problem of securely pairing two devices that do not share prior secret key, or have each other's authentic public key. One of the main challenges in secure pairing, however, is to provide usability while guaranteeing security.

Commodity wireless solutions such as Bluetooth or Wi-Fi have standards that attempt to provide a secure exchange of credentials (e.g., Bluetooth Secure Simple Pairing (SSP) [64] and Wi-Fi Protected Setup [22]). We illustrate as an example the details of one of the SSP protocols called *numeric comparison*. This protocol consists of two phases in performing a secure pairing. In the first phase, a pair of devices exchange their public keys (e.g., Diffie Hellman Key Exchange). Then in the second phase, both devices perform verification on the received public keys by requiring the user to verify if the displayed numbers on both devices are identical. Once the user performs a successful verification, the devices then establish a secure connection. However, Kuo et al. [85] highlight that large attack surfaces for these specifications exist, and provide recommendations to improve usability. For example, the security of the *numeric comparison* method depends on the displayed number to be hard to guess, and not repeated. However, in many products, manufacturers are not careful in

implementing the standards, and cause potential security vulnerabilities.

Many researchers propose different solutions to achieve secure pairing while preserving usability. They attempt to leverage different types of OOB channels to provide demonstrative identification. First, researchers propose using visual channels as the OOB channel. McCune et al. propose Seeing-is-Believing (SiB) [96], a solution that allows two smartphones to securely exchange each other's public keys using QR codes and phone cameras. SiB, however, is not well suited for a vehicular setting because it requires extra hardware such as cameras, which is not present in vehicles. SiB also requires user diligence as the users need to actively take pictures of the QR code.

There are solutions that explicitly require human users to play the role of the OOB channel. Gehrmann et al. [58] propose a series of protocols named Manual Authentication (MANA) by allowing the user to read and input short strings to different devices. Although MANA protocols may seem secure, we claim that MANA is not suitable for solving our problem because it requires user diligence which often leads to human errors, eventually resulting in possible attacks.

Many researchers also investigate the usability of different secure pairing proposals [77, 83, 131]. The results of these studies conclude that authentication mechanisms that involve active user participation (e.g., comparing numbers) is one of the most important factor in influencing user's perception of the convenience of a specific pairing solution.

3.7 Discussion

We now discuss some of the relevant points that were not addressed in the above sections.

Alternative Pairing Methods: There are alternative pairing solutions that may seem to be valid at a first glance for performing a secure key agreement. However, we provide reasons for why they may not be adequate solutions. First, many cars are already equipped with built-in iPod jacks. While it is possible to perform secure key agreement using such cables, we find that not all existing cars today have such cables. We design *MVSec* to be deployed in all cars, including existing cars without such cables. Second, NFC may be used as an OOB channel to perform authentication. However, NFC suffers the same issue – not all cars are equipped with NFC chips today. Furthermore, many smartphones and tablets including all iOS devices ship without NFC chips. We find that NFC cannot meet our goal of deploying *MVSec* to all existing cars, while incurring minimal hardware cost.

Visual Channel: Recall that our solution leveraging visual channel was established by varying the light intensity in the glove compartment to emit signals to the phone inside the compartment. While current cars today only have a simple mechanical controller that turns on the light when the compartment door opens, we envision that the light source can be controlled by either installing a new ECU (Electronic Control Unit) or being controlled by existing ECU in the future. To support *MVSec* in existing cars, dealers can easily service existing cars to install such controllers.

Access Control Policy: *MVSec* employs an access control policy where the right to drive the vehicle equates to the right to pair a phone. In addition, the driver may delegate such rights to the passengers. However, there may be situations that such policy may not be sufficient. This is best exemplified when the driver leaves his car with valet parking or repair service center. If the glove compartment is unlocked, the valet or service personnel may pair their phones with the car. To resolve this issue, we envision *MVSec* to employ the following mechanism. *MVSec* may enforce the car to prompt the driver's phone for any additional pairing requests, so that the car would only proceed with the pairing process after the driver's authorization. We assume that first phone to be paired does not require such authorization.

Bluetooth Discovery Overhead: *MVSec* aims to increase usability while guaranteeing security. One of the main usability drawbacks of the current Bluetooth pairing process is the slow device and service discovery phases. In addition to using the OOB channel for security, we may leverage the OOB channels to transmit the vehicle's Bluetooth MAC address, eliminating human efforts in Bluetooth pairing setup. Thus, the only task the user has to perform is clicking a button on a vehicle and a smartphone. The task will take less than 13 seconds to transit a 48-bit long Bluetooth mac address using sound signals in the current implementation.

Potential OOB Channels in Future: We find that newer cars will be equipped with various types of sensors and actuators, which can be used to establish alternative OOB channels. For example, the vehicular industry is moving towards installing haptic seats (using vibrators) and accelerometers. These new equipment can help establishing an alternative weak OOB channel with a smartphone, as the phone can receive messages encoded via the vibration signals using its accelerometer, and also transmit its signal encoded using vibration signals as well.

3.8 Chapter Summary

Wireless device pairing is often vulnerable to MitM attacks. Thus, secure pairing between a vehicle and a phone is important for a successful industry deployment. The proposed protocols in this chapter address solutions to protect against these attacks, while providing demonstrative identification to the human user. *MVSec* leverages readily available hardware to allow a car and a phone to perform secure key agreement without any pre-shared secret, and independent of a trusted third party, while still preserving usability.

Chapter 4

Less Constrained: Smart Home

We now present an application scenario that is less constrained than that of the previous scenario, namely autonomous IoT device pairing in a smart home. This is less constrained in terms of physical boundary, because a single detached house inherently provide a physical separation between devices inside and outside. However, unlike the previous scenario, wireless signals as well as some attenuated sound or vibration signals still propagate to the devices outside of the boundary.

4.1 Problem Definition

Securing the IoT network becomes a necessity as introducing more IoT devices comes at a cost of potential privacy leakage as these devices are equipped with sensors that monitor activities within a house [70, 100, 107]. While securing the connectivity of the IoT devices is crucial, it is non-trivial for an end user to perform security configurations on the devices. We refer to performing security configurations (e.g., configuring Wi-Fi WPA2 for secure pairing) in this chapter as establishing a secure channel between devices (i.e., cryptographic key agreement). First, many IoT devices come without any I/O mechanisms. Trying to configure a motion detector without a proper display and keypad will be extremely cumbersome for home owners. Second, there are too many IoT devices to configure. One can already easily find tens of IoT devices in a home. Furthermore, the number of IoT devices in a smart home are even projected to increase to 500 within the next decade [47]. To exacerbate the problem, the devices may need multiple security reconfigurations during the life span of these devices. Such reconfigurations may be due to multiple reasons – new cryptographic algorithm may be introduced requiring longer keys, or the IoT devices may themselves be compromised and

the keys may have been breached – all of which, require re-establishment of cryptographic keys. In addition to the aforementioned I/O problems, many of the devices, such as smoke detectors, HVAC systems, etc., are permanently installed and are extremely cumbersome to physically reinstall after reconfigurations. Considering a long life span of many of these devices, which usually range in the order of years, security reconfiguration introduces a serious usability drawback. While newer IoT devices are starting to be equipped with NFC or other out-of-band channels such as light to help in solving the I/O problem, such solutions still do not scale with many devices requiring multiple reconfigurations and additional specialized hardware [13, 14].

The aforementioned drawbacks necessitate an autonomous pairing mechanism for the IoT devices that require no human involvement in any of the pairing processes. To address this concern, we propose *Perceptio* (which means perception in Latin), an autonomous pairing scheme based on the similarity of contextual information collected by sensors of the IoT devices. *Perceptio* makes use of the findings that IoT devices co-located within a physical boundary such as a room will experience more of the similar events over time as opposed to potential devices owned by an attacker, who may be outside of the physical boundary. Specifically, physical boundary such as homes are naturally enforced with a notion of physical security. For example, access rights to an apartment complex are only granted to the residents. Visitors are also granted access after an implicit or explicit delegation of trust. In this section, we are inspired by such notion of trust within a physical boundary because it provides a natural separation of trust. Hence, we envision devices within a physical boundary would also trust each other compared to devices that are outside of an apartment. Figure 4.1 depicts this idea where the user's sensors observe the events occurring within an apartment, while the attacker who is outside cannot observe such events.

However, many challenges exist to fully address the aforementioned problem. First, we identify new challenges related to devices that have disparate sensing capabilities. We find from our main observations of recent trends in commercially available IoT devices, which reveals the emergence of special-purpose sensing devices with only a small number (often one) of embedded sensors, typically optimized for a specific application for cost, power consumption, and form factor. For example, Passive infrared motion detectors are only equipped with a single infrared sensor to monitor movements [35]. Hence, it is infeasible to directly compare context collected from different devices with different sensor modalities because differing sensor types produce completely different signals. Toward such goals, we need to gain a stronger understanding of the contextual content of sensory data as observed

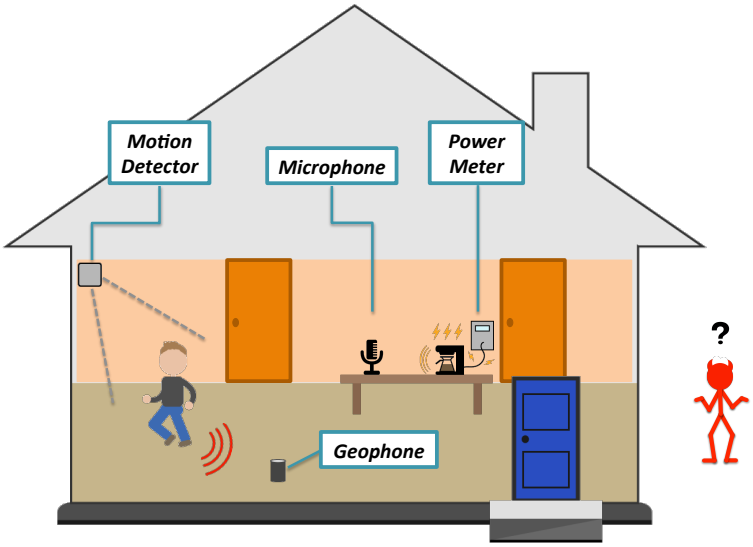


Figure 4.1: A physical boundary (house) provides a perceptual separation between user's devices inside vs. other devices outside, enabling *context-based* pairing via observations of random events within the house.

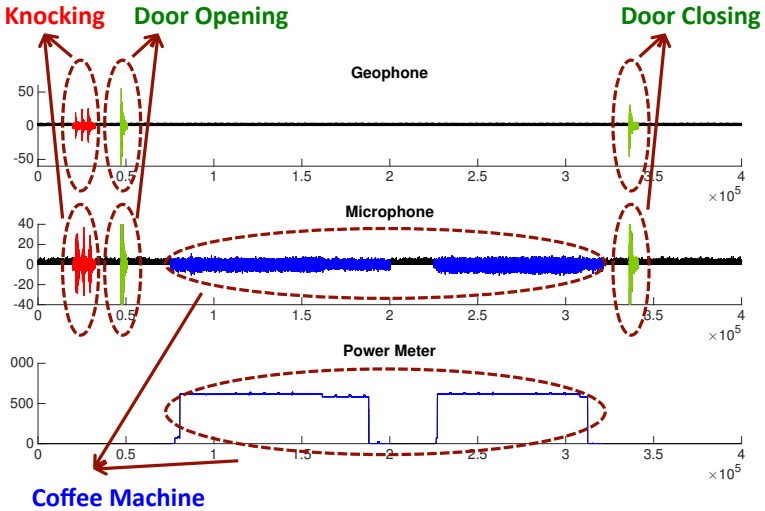


Figure 4.2: We demonstrate how different types of sensors are capable of measuring aspects of the same events.

from different IoT devices. To do this, we can gain some insight from analogous human behavior through the following scenario. Suppose that one person with a hearing impairment and another with a visual impairment are both in a room. When the door to the room closes, both people can observe the event at the same time, but using different senses; the hearing impaired person can see the door closing, while the visually impaired person can hear the

door closing. Because of the timing, both people could share their observations and determine they had witnessed the same event. This analogy can be further extended to include events that humans perceive in multiple ways. For example, we perceive rainfall through hearing, feeling, and seeing raindrops [28]. By applying this analogy to the IoT device space, we can similarly leverage timing information as an invariant property among heterogeneous devices. We thus develop our approach using a principle we refer to as "*numerically different yet contextually similar*" observation of events, exploiting commonly observed timing information. In the IoT device regime, we provide a more detailed example to demonstrate the ability for disparate sensing devices to measure common events. In this scenario, Bob knocks on his roommate Dan's door to invite him for coffee in the living room. Dan opens his bedroom door and walks into the living room, and Bob then makes two cups of coffee. After enjoying their coffee together, Dan goes back into his bedroom and closes the door. Suppose now that Bob and Dan have deployed IoT devices with a *geophone* and *microphone* and that the coffee machine is connected to a *power meter*. In this case, the corresponding sensor readings from these devices capture the events, as depicted in Figure 4.2. The different types of sensors are capable of perceiving some events in common. In particular, the geophone and the microphone both capture the knocks and door opening/closing events, while the microphone and power meter both capture the activity of the coffee machine.

Hence, we make use of the findings that many sensors of different modalities actually respond to and perceive the same context. Specifically, *Perceptio* leverages a common feature across different modalities, namely the time intervals of starting points of the common events. Because *Perceptio* leverages the time intervals, it does not require the devices to be time synchronized.

Yet, challenges still remain, as different sensors located at different positions within a room would produce similar but not equal signals. Hence *Perceptio* tolerates these error by leveraging a fuzzy commitment scheme, which bases its error tolerance on Reed-Solomon error correction mechanism.

To evaluate the design of *Perceptio*, we perform experiments by equipping a room with a variety of sensors to represent existing and prevalent commercial IoT products. Our deployment includes a microphone (smart speakers [23, 61]), an accelerometer (on-object sensors [56, 104, 123]), a motion detector [52, 102], a power meter [72, 74], and a geophone (structure or footstep monitors [107, 130]). In addition, we deploy corresponding devices as well as higher quality microphone and accelerometer outside the room to represent the attacker's devices. Human participants perform a number of typical events in the room,

providing the ambient inputs to the various sensors. As a proof of concept, our empirical evaluation demonstrates that fingerprints generated by devices within the room are far more likely to match (yielding an average of 94.9%), while the highest fingerprints generated by the attacker's devices outside the room have low similarity to those inside the room (only yielding an average of 68.9%). To support the proof of concept, we study existing data sets for activity within smart homes to quantify the available entropy and the corresponding amount of time for devices to establish keys with sufficient confidence.

4.2 System Models

This section presents our threat model describing the goals and capabilities of the attacker. Subsequently, we present the assumptions and constraints of *Perceptio*.

4.2.1 Threat Model

The goal of the attacker is to *leak private information* of home occupants by eavesdropping on the communication between IoT devices. In order to achieve this goal, the attacker may launch (1) *Shamming attack* or (2) *Man-in-the-Middle attack*.

We define a Shamming attack where the attacker's device, $\mathcal{M}$ (placed outside of the house but within the wireless communication range), succeeds in fooling a legitimate device, $\mathcal{LD}$ (inside the house), to accept the pairing as another $\mathcal{LD}$. $\mathcal{M}$ may launch three types of Shamming attacks. First, it may launch an *(1-a) Eavesdropping attack* by attempting to sense (from outside) the events occurring inside. $\mathcal{M}$ may have following three levels of capabilities to launch this attack. $\mathcal{M}$ may have *(i) normal-level of resources* equipped with standard off-the-shelf IoT sensors that are comparable to $\mathcal{LD}$s inside the house. $\mathcal{M}$ may also have *(ii) medium-level of resources* equipped with higher-end off-the-shelf consumer electronic devices that are more powerful than (i). Furthermore, $\mathcal{M}$ may have *(iii) powerful-level of resources* equipped with asymmetric capabilities (e.g., military-grade thermal imaging and x-ray vision). As such, we focus on (i) and (ii) and disregard (iii) because such attackers could already visualize activities within the home and reveal private activities, independent of *Perceptio* and the IoT devices deployed within the home. Moreover, the attacker may launch other types of Shamming attack such as: *(1-b) Signal Injection attack* – by creating events with large noise or vibration from outside (e.g., using jack-jammer); or *(1-c) Sensor Spoofing attack* – by injecting spoofing signals to $\mathcal{LD}$s. The attacker launches either of these

attacks again in an attempt to allow both $\mathcal{M}$ and $\mathcal{LD}$s to perceive simultaneous event signals and ultimately succeed in fooling $\mathcal{LD}$s to accept the pairing with $\mathcal{M}$.

Second, $\mathcal{M}$ may launch a man-in-the-middle (MitM) attack on key agreement messages between a pair of $\mathcal{LD}$s by simply intercepting messages transmitted over the wireless medium. Such an attacker is able to use a variety of primitives such as injection, replay, modification, and blocking/deleting messages in the communication channel.

4.2.2 Assumptions and Constraints

We assume that the physical boundaries of a house draw a natural trust boundary for deployed devices, $\mathcal{LD}$s. This assumption reflects scenarios in which $\mathcal{LD}$s inside the boundary are owned and operated by a common entity (e.g., home owner). However, non-authorized personnel do not have access to the physical space, hence do not have control over the IoT devices. We also assume that the family members and authorized guests are not malicious. For example, if one's family members or authorized guests are the only people who have access to their house, and devices brought into the home for prolonged periods of time are assumed to be trustworthy, then a proof of deployment within the house is sufficient to bootstrap a trusted connection to the IoT network. We view the introduction of unauthorized devices into the home by malicious guests as a problem of the homeowner's physical security, not as a relevant problem of secure pairing. Hence, this issue is out of scope for our work.

In addition, we acknowledge that single-family homes are made up of a number of joined rooms, and the separating walls actually present numerous physical boundaries within the home. While sensors within the same home are likely to perceive some common events due to the common physical structure, the walls are bound to induce a *non-negligible attenuation factor*, with different propagation media causing distortion and attenuation of mechanical signals. More specifically, walls and joints are known to cause material damping, reflection and diffraction of acoustic and vibration signals [59, 75]. However, since interior walls tend to provide far less attenuation compared to exterior walls, we expect a fair amount of signal to propagate between nearby IoT devices, at least a sufficient amount to allow for IoT network connectivity, as full pairwise connectivity is likely unnecessary. As we will discuss later, it may also be possible to configure a small number of IoT devices to act as "bridging devices", if needed, to facilitate secure pairing across the internal walls of the home.

In either case, we design *Perceptio* to rely on the core observation that sensors outside the home cannot *consistently perceive* the relevant activities inside with *similar fidelity* as

$\mathcal{LD}$s. While our design focuses on single-family detached housing (comprising 61.5% of U.S. housing [95]), we believe that future extensions of *Perceptio* could extend our work to other multi-tenant attached housing (e.g., apartments or townhouses) through rigorous engineering of thresholds and other protocol parameters.

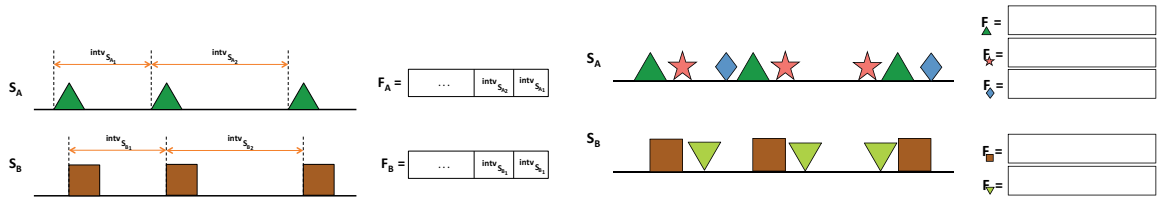
4.3 Capturing Contextual Cues

As with any cryptographic key agreement protocols, *Perceptio* needs to bootstrap its trust from a source of entropy. We leverage the inherent randomness of events occurring in a room (e.g., knocking, walking, talking, etc.) as its source of entropy in its cryptographic protocol. Specifically, *Perceptio* leverages the fact that it is infeasible for an attacker to guess a series of event occurrence in sub-second granularity. Hence, fingerprint extraction from contextual information is a vital part of *Perceptio*. We now discuss the design choice of the fingerprint extraction algorithm, and how multiple event types affect *Perceptio* fingerprinting.

We now address the seemingly impossible challenge of trying to "fingerprint" disparate sensor modalities across IoT devices. This is possible because sensors of different modalities "perceive" the same context even though their numerical representation may be different. Hence, *Perceptio* abstracts out from differing numerical sensor data, and leverages temporal domain as common feature across the devices. Specifically, *Perceptio* makes use of the fact that starting points of commonly observed events are spaced out at equal time intervals, and captures the collection of these time intervals as fingerprints. Figure 4.3(a) depicts an example of how two sensors of different modalities fingerprint a commonly observed context. Assuming that both $Sensor_A$ and $Sensor_B$ observed an event, their numerical representations of the event are shown as *triangles* and *circles*, respectively. Note that intervals between the starting times of adjacent *triangles*, denoted as $intv_{S_{A_1}}$ and $intv_{S_{A_2}}$, while those of *circles* are denoted as $intv_{S_{B_1}}$ and $intv_{S_{B_2}}$. *Perceptio* makes use of the fact that $intv_{S_{A_i}}$ and $intv_{S_{B_i}}$ are very similar to each other. Subsequently, the intervals are converted into bits and appended to the fingerprints as:

$$\begin{aligned} F_A &= \{intv_{S_{A_n}}||intv_{S_{A_{n-1}}}||...||intv_{S_{A_1}}\} \\ F_B &= \{intv_{S_{B_n}}||intv_{S_{B_{n-1}}}||...||intv_{S_{B_1}}\} \end{aligned} \tag{4.1}$$

However, we need to address an additional challenge of how disparate sensors have varying events (e.g., walking, door opening, talking, etc.) that they are responsive to. For example,



(a) Single event commonly observed

(b) Events observed by each sensor

Figure 4.3: Creating F with starting point intervals.

consider $Sensor_A$ and $Sensor_B$ of different modalities, and the corresponding events that the two sensors observe. Events that only $Sensor_A$ and $Sensor_B$ observes are denoted as $E_A\overline{E_B}$ and $\overline{E_A}E_B$, respectively. Events that both modalities observe in common are denoted as E_AE_B. For example, a microphone will be more sensitive to people speaking, geophones will be more sensitive to footsteps of a walking event, and both of these sensors are responsive to a running coffee machine (i.e., $E_{mic}\overline{E_{geo}} = \{talking\}$, $\overline{E_{mic}}E_{geo} = \{walking\}$, and $E_{mic}E_{geo} = \{running\ coffee\ machine\}$). However, the core idea of *Perceptio* lies in the fact that most pairs of disparate sensor types have sets of common events that they respond to with varying sensitivity.

In Figure 4.3(b), $\{\blacklozenge, \blacktriangle, \bigstar\}$ and $\{\blacktriangledown, \blacksquare\}$ depict set of signals that are observed by $Sensor_A$ and $Sensor_B$, respectively. Hence, each sensor will first locally determine similar events (i.e., clustering events into different clusters), and extract corresponding fingerprints per event type. Hence $Sensor_A$ will have three distinct fingerprints (i.e., $F_{\blacklozenge}$, $F_{\blacktriangle}$, and $F_{\bigstar}$), while, $Sensor_B$ will have two distinct fingerprints (i.e., $F_{\blacktriangledown}$ and $F_{\blacksquare}$).

4.3.1 Fingerprint Entropy

Perceptio bootstraps its trust from the entropy of the occurrence of captured events. These occurrences are converted to the intervals of the starting points, which in turn are translated into the bits of the fingerprint. Hence, the entropy of the fingerprint depends on the length of the fingerprint. This is depicted in Equation 4.2. F depicts the concatenation of bit values of each intervals, $intv_{S_{A_i}}$, for $i = [1, n]$ intervals, where $[1, n]$ represent integers from 1 to n, inclusive. If the length of F are greater than l_{min}, a minimum length of a fingerprint, F is truncated to l_{min} bits. If less than the minimum length, the fingerprint is discarded due to lack of enough entropy.

$$F_{E_A} = \begin{cases} \Big[F\Big]_{l_{min}}, & \text{if } |F| \geq l_{min} \\ \emptyset, & \text{otherwise} \end{cases} \tag{4.2}$$

4.4 Design and Implementation

This section presents *Perceptio* protocol and its design and implementation details.

4.4.1 *Perceptio* Protocol Details

Perceptio's fingerprint verification incorporates the fingerprint, F, into a cryptographic protocol to yield a verifiable shared symmetric key between the two parties. Figure 4.4 depicts the high-level overview of *Perceptio* protocol. (1) Initially two devices with disparate sensor modalities captures numerically unequal time series data streams. (2) While co-located devices observe similar events, the extracted pair of fingerprints will not be exactly the same due to sensitivity and different modalities. (3) We treat such subtle differences in fingerprints as errors and tolerate them using a fuzzy commitment scheme [51, 76] building on error correcting codes. (4) Finally two devices share a master symmetric key, k, and can subsequently generate shared session key, k_{AB}. Similar to the related work [68, 99], we design a *Key Strengthening Process*, which gradually *strengthens* the initially shared (but potentially insecure) key. This is made possible by gradually increasing the authenticity confidence over time through repeated execution of the fuzzy commitment using different fingerprints (Steps (1) through (4)), until a minimum confidence score is attained, inherently making it extremely difficult for Shamming attacker devices (located outside of the physical boundary) to sustain the shared key.

In the *Key Agreement Phase*, A and B generates fingerprints $\{F_{A_i}, i = 1, \ldots, p\}$ and $\{F_{B_j}, j = 1, \ldots, q\}$ for the p and q observed event clusters. Device A then encodes a randomly generated secret key k_i using each fingerprint F_{A_i}, $i = 1, \ldots, p$, to create a set of commitments as $C_{A_i} = F_{A_i} \ominus ENC(k_i)$, where $\ominus$ is subtraction in a finite field, $\mathbb{F}^n$, equivalent to an XOR operation, and $ENC(\cdot)$ is the encoding operation for an error correcting code (e.g., Reed-Solomon). A then sends $\{(C_{A_i}, h(k_i)), i = 1, \ldots, p\}$ to B, where $h(\cdot)$ is a collision-resistant hash function, which discloses no information about the keys k_i or the fingerprints F_{A_i}. Upon receiving the set of commitments from A, device B attempts to open the

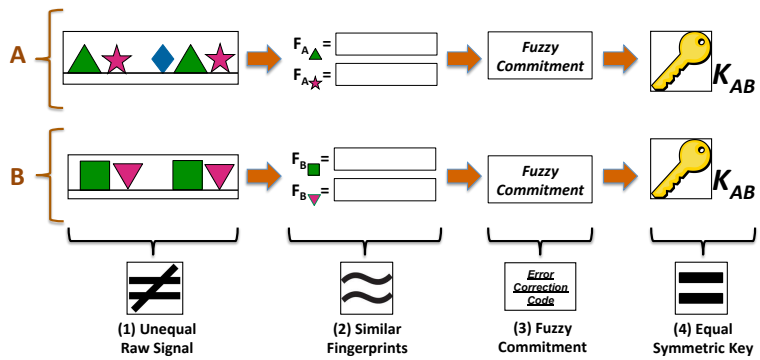


Figure 4.4: Figure depicts *Perceptio* protocol overview. Unequal heterogeneous sensors data from A and B are eventually converted to numerically equivalent symmetric key.

commitment to acquire any one of the original secrets k_i using its fingerprints F_{B_j}. B computes $\hat{k}_{i,j} = DEC(F_{B_j} \ominus C_{A_i})$ for all i, j pairs, where $DEC(\cdot)$ is the complementary decoding function, such that $DEC(ENC(m) \ominus \nu) = m$ for a bit string m whenever the Hamming weight (l_1 norm) $|\nu|_1$ is within the code's decoding capability t. If B finds an i, j pair such that $h(k_i) = h(\hat{k}_{i,j})$, then it most likely found a fingerprint match,$F_{A_i} \approx F_{B_j}$. There are many protocol variations at this point, but we choose one in which B needs to find only one such pair, so not all pq values need to be computed if a match is found. At this point, B can use a key derivation function $KDF(\cdot)$ [111] to create a shared symmetric key as $k_{AB} = KDF(\hat{k}_{i,j})$, though A is unaware of this key at this point (Figure 4.5 Steps 1-4).

To allow A to generate the matching symmetric key k_{AB} and verify it actually matches the key generated by B, both A and B further participate in the *Key Confirmation Phase*. B generates a random nonce n_B and transmits β, where $H(\hat{k}_{i,j})$ equals to $H(k_i)$ and $M_k(m)$ represents a keyed message authentication code (MAC) of message m using key k. A, upon receiving this message, first identifies the key, k_i, from $H(k_i)$. If found, A derives the shared key as $k_{AB} = KDF(k_i)$ for the matching i. A then performs a MAC verification with k_{AB} and if successful, it also generates a nonce, n_B, and transmits to B, α. B, upon receiving α, performs MAC verification to verify that A also generated the same key k_{AB}. If successful, device A and B successfully computed a shared symmetric key for one round (Figure 4.5, Steps 5- 8).

In addition to the aforementioned protocol, *Perceptio* includes an optional extension to allow a notion of *transitive verification* for cases where two devices want to verify each other but their sensing equipment does not allow for generation of matching fingerprints (e.g., the accelerometer and the power meter who perceive no event in common). This is synonymous

Key Agreement Phase

1. A : $F_{A_i} = extractFs(ctx, t_F)$; $i = 1, \ldots, p$
   B : $F_{B_j} = extractFs(ctx, t_F)$; $j = 1, \ldots, q$

2. A : $k_i \xleftarrow{R} KGen(1^\gamma)$
   $C_{A_i} = F_{A_i} \ominus ENC(k_i)$

3. $A \rightarrow B$: $C_A = C_{A_1}||H(k_1), .., C_{A_p}||H(k_p)$

4. B : $\hat{k}_{i,j} = DEC(F_{B_j} \ominus C_{A_i})$
   Verify $H(k_i) \stackrel{?}{=} H(\hat{k}_{i,j})$; Aborts if fails
   Creates $\hat{k}_{AB} = KDF(\hat{k}_{i,j})$

Key Confirmation Phase

5. $B \rightarrow A$: $\beta = H(\hat{k}_{i,j})||n_B||M_{\hat{k}_{AB}}(n_B)$,
   where $n_B \xleftarrow{R} \{0,1\}^\eta$

6. A : Creates $k_{AB} = KDF(k_i)$;
   $M_{\hat{k}_{AB}}(n_B) \stackrel{?}{=} M_{k_{AB}}(n_B)$; Aborts if fails

7. $A \rightarrow B$: $\alpha = n_A||M_{k_{AB}}(n_B||n_A)$,
   where $n_A \xleftarrow{R} \{0,1\}^\eta$

8. B : $M_{k_{AB}}(n_B||n_A) \stackrel{?}{=} M_{\hat{k}_{AB}}(n_B||n_A)$;
   Aborts if fails

Figure 4.5: Details of *Perceptio* key agreement and confirmation protocol using contextual information

to a disconnected graph if the nodes are sensors and edges are commonly perceivable event. We call this extension ***Transitivity of Trust (ToT)***. If the two devices A and C have each performed the fingerprint verification with a third device B, meaning A and B share key k_{AB} and B and C share key k_{BC}, A and C can rely on ToT to expand the "pairing" operation to a "grouping" operation by leveraging authenticated encryption scheme [115] to exchange public parameters for Diffie-Hellman key exchange [50]. Furthermore, this approach enables devices located in different rooms within a house to pair, leveraging *bridging devices*. We discuss this extension further in Section 4.7.

4.4.2 Implementation

Figure 4.6 depicts the flow chart diagram of fingerprint generation steps. First a sensor captures contextual information for fingerprint time period, t_F, producing a raw sensor data. This is first input to *Signal Detection* module, which distinguishes signals of events (e.g., walking, talking , etc.) against ambient noise and outputs the corresponding indices of the event signals. Subsequently, these indices, along with the raw sensor data, are input to *Event Clustering* module, which performs unsupervised learning to cluster signals of similar events via K-Means clustering. Hence, module outputs different cluster IDs and the corresponding indices of the signals belonging to the clusters. The output is then input to *Fingerprint Extraction* module, which finally converts the cluster indices into fingerprints to be used in *Perceptio* protocol. We now present the implementation details of *Signal Detection* and *Event Clustering* modules.

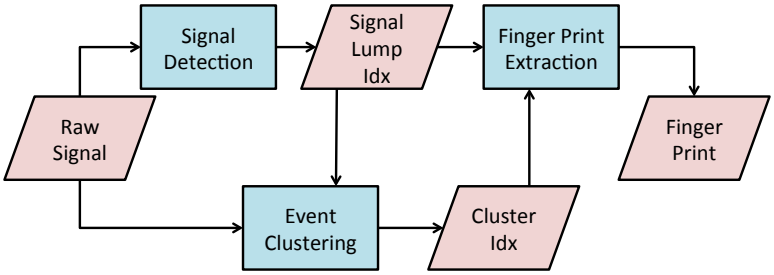


Figure 4.6: Overview of *Perceptio* fingerprint generation flow chart.

4.4.2.1 Signal Detection

The goal of signal detection module is to identify the signals that represent events of interest as opposed to ambient noise. We break down the tasks into two steps – (1) performing a moving average; and (2) thresholding and signal detection. We first compute a moving average to smooth out the signal for noise removal using *Exponentially Weighted Moving Average (EWMA)*. Figure 4.7(a) illustrates this effect, as the first plot depicts the original geophone signal of the event of a person walking. The second plot depicts the absolute value of the original plot, and the third plot depicts the *EWMA* of the absolute value.

We then perform *thresholding* for signal detection. We note that we have two types of thresholding, namely a lower-bound (Thr_{lower} and an upper-bound (Thr_{upper}) threshold. We leverage Thr_{lower} to distinguish event signals to ambient noise. On the other hand, we leverage Thr_{upper} to remove any signals of high amplitude, in order to thwart an eavesdropping attack.

We note that Thr_{upper} can be a function of Thr_{lower} after certain calibration phase. This is depicted in Figure 4.7(b) (a), where we apply a lower-bound thresholding to the *EWMA* signal using the lower dotted line (i.e., $Thr_{lower} = 3$). The signal above the threshold are highlighted with a gray box. Also, we apply an upper-bound thresholding as well using the upper dotted line (i.e., $Thr_{upper} = 10$). For more accurate event clustering, however, we implement a signal *lumping* technique to group segmented parts of the event signal into a single event signal, as shown in Figure 4.7(b) (b). Specifically, we disregard short discontinuities between adjacent segmented signals above threshold to "lump" the signals into one continuous group of signal event. From the indices returned by these steps, we determine the signal of interest in the original signals as presented in Figures 4.7(b) (c) and (d), depicting before and after lumping technique, respectively. Finally, this module outputs the corresponding indices of detected signal to the *Event Clustering* module.

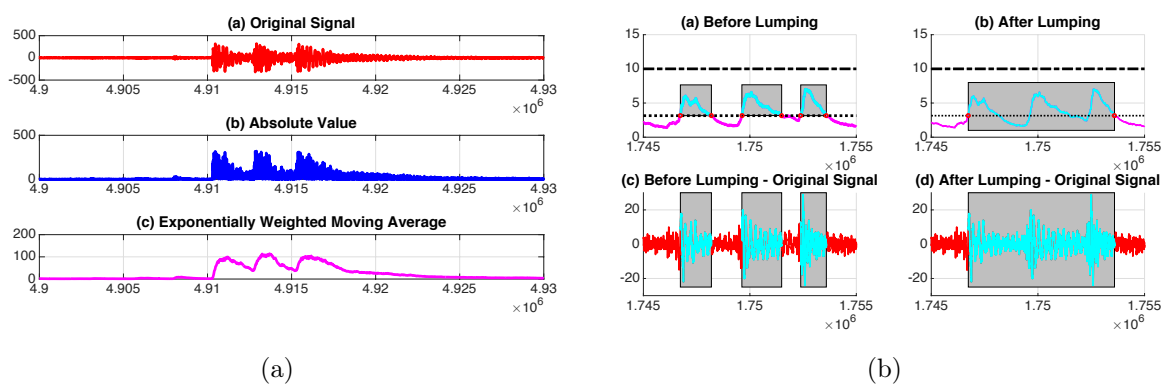
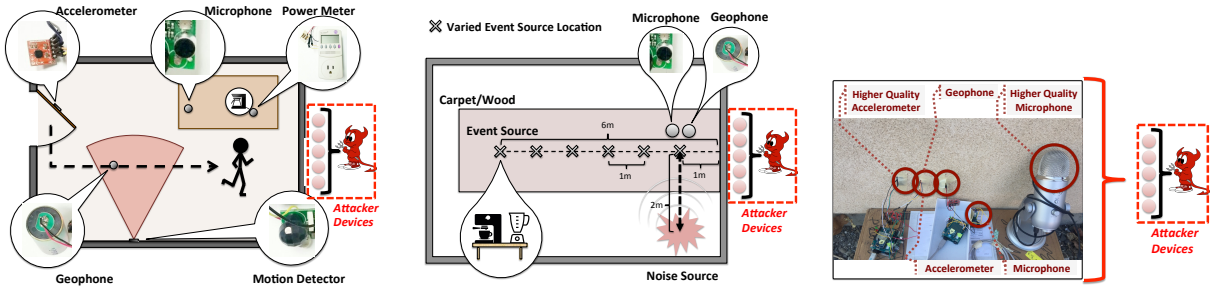


Figure 4.7: Illustration of (a) raw geophone signal, followed by corresponding absolute value, and subsequent exponentially weighted moving average; (b) thresholding and signal isolation.

4.4.2.2 Event Clustering

We implement event clustering to reliably categorize observed events into appropriate cluster groups. Though some additional work may increase the accuracy and efficiency of the clustering results, we present a preliminary proof-of-concept implementation details.

We select a set of features per sensor to reliably separate perceived events via clustering such as maximum amplitude, duration, and area under the curve and its variants. We leverage K-Means Clustering to cluster data points of similar groups, without prior training, by taking in hypothesis cluster number, K. Hence, we make use of Elbow method to infer the optimum value of K [79], which tests several number of K cluster hypothesis to find the optimal K



(a) Evaluating event detection accuracy for various sensing modalities

(b) Evaluating the impact of different environmental factors

(c) Attacker devices

Figure 4.8: To study event detection accuracy for $\mathcal{LD}$s and $\mathcal{M}$s of different sensor modalities, we have human subjects conduct the following actions shown in (a): knock on a door hosting an accelerometer, walk across a motion detector, around a microphone and geophone on the ground, and brew coffee from a machine attached to a power meter. The attacker sensors are placed outside the wall opposite to the door. We study the effect of environmental factors in (b): a coffee machine and blender are used successively while varying the distance between them and the sensors, the floor type and the noise level inside the room. We illustrate the five $\mathcal{M}$s in (c) including higher quality accelerometer and microphone.

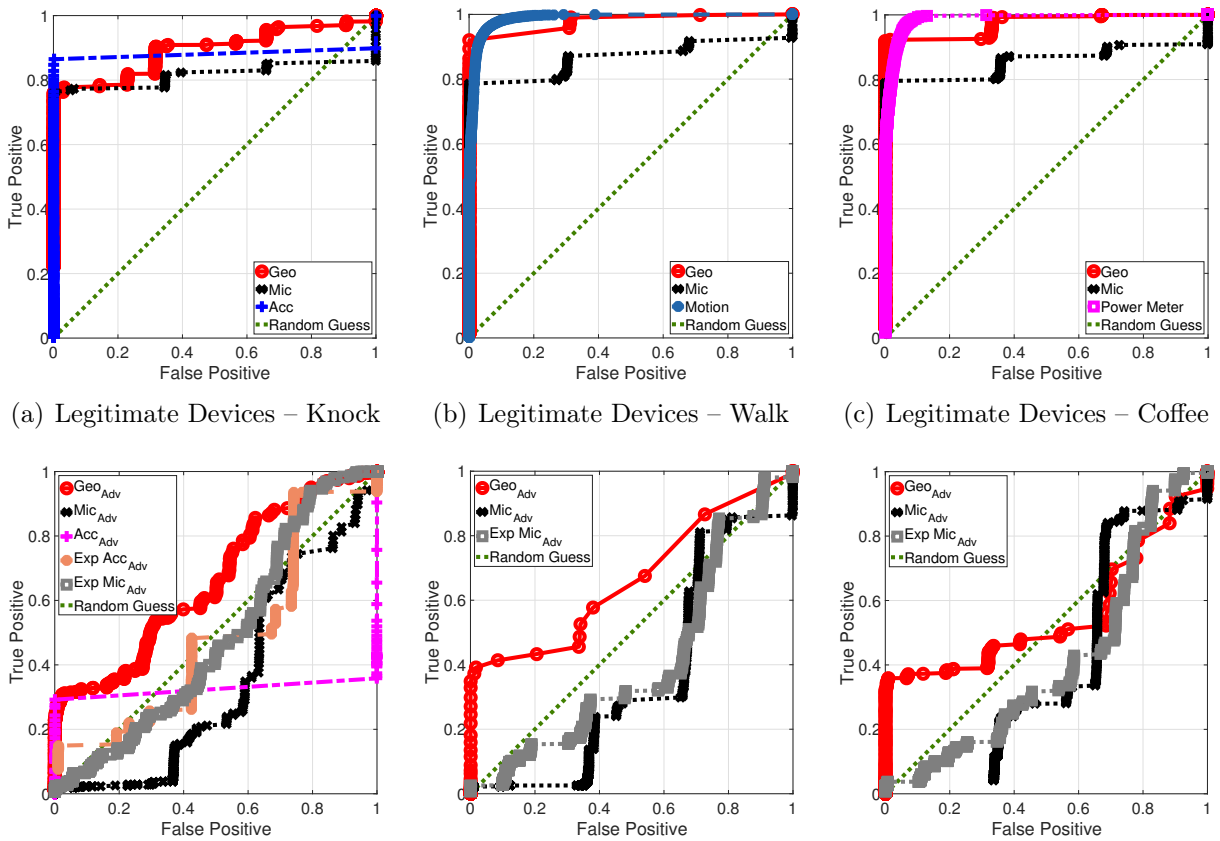
value. Consequently, K-means clustering and elbow method enables each sensor device to roughly group perceived events to categories without the burden on the manufacturers to train specific events for all the devices.

4.5 Evaluation

We implement the *Perceptio* protocol and evaluate its effectiveness in different settings. After detailing the apparatus used, we present an end-to-end study of *Perceptio*'s various aspects, including sensors' event detection abilities and robustness of fingerprint similarity and key establishment.

4.5.1 Experiment Apparatus

We describe the nature of legitimate devices, $\mathcal{LD}$s, placed inside the environment and attacker devices, $\mathcal{M}$s, placed outside attempting to launch Shamming–Eavesdropping attack. The $\mathcal{LD}$s include a SM-24 geophone [38], an MD9745APA-1 microphone [21], an ADXL335 accelerometer [49], an MP Motion Sensor NaPiOn passive infrared motion detector [108], and a Kill-A-Watt P4400 power meter [74]. Each of the sensors is interfaced to an Arduino



(a) Legitimate Devices – Knock (b) Legitimate Devices – Walk (c) Legitimate Devices – Coffee

(d) Attacker Devices – Knock (e) Attacker Devices – Walk (f) Attacker Devices – Coffee

Figure 4.9: We study the ROC of $\mathcal{LD}$s and $\mathcal{M}$s for accuracy of event detection. Across all events, the $\mathcal{LD}$s have a high detection rate while the $\mathcal{M}$s (even the higher-quality microphone and accelerometer) hardly perform better than a random guess. (Note: For each event type, we only show sensors whose modalities have the ability to detect that event. For example, the accelerometers cannot detect the coffee machine, hence are ignored in (c) and (f)).

Uno board [10] with a Wireless SD Shield [19] and microSD card for data logging at 5 kHz sampling rate. The sensors were placed between 2.5-5.5m apart from each other. The $\mathcal{M}$s also include a SM-24 geophone, MD9745APA-1 microphone, and an ADXL335 accelerometer, as well as a higher-quality MMA1270KEG accelerometer [118] and a higher-quality Blue Yeti microphone [98] as depicted in Figure 4.8(c). The higher-quality accelerometer and microphone cost an estimated \$10 and \$100 respectively, which is roughly one and two orders more expensive than the normal-quality IoT accelerometer and microphone.

4.5.2 Event Detection

4.5.2.1 Detection Abilities of Legitimate and Attacker Devices

We now evaluate the performance of each sensor in distinguishing event signals from ambient noise. Recall from Section 4.4.2.1 the three variables of interest are a lower-bound threshold Thr_{lower} to separate the signal from noise; an upper-bound threshold Thr_{upper} to discard distinct signals with high amplitude to thwart Shamming–Eavesdropping attacks; and the weight α used in the exponential moving average. In this experiment, we vary Thr_{lower}, which is important in signal detection, while fixing Thr_{upper} and α to empirically optimized values.

We illustrate the study setup in Figure 4.8(a). The experiment is conducted in a squash court wherein the $\mathcal{LD}$s are arranged with the geophone on the floor, the microphone on a table, the accelerometer on the door, the motion detector aimed at the center of the room, and the power meter supporting a single serving coffee machine (Nespresso Pixie Carmine [101]). The $\mathcal{M}$s deployed just outside the room (as illustrated in Figure 4.8(a)) include the accelerometer, the higher-quality accelerometer and the geophone attached to the outside of one of the walls of the squash court and the microphone and higher-quality microphone placed on the ground adjacent.

We have ten human subjects perform the following tasks: knock on the door hosting the accelerometer, walk across the court (across the motion detector and the geophone) and around the table, brew coffee from the espresso machine on the table two times, one after another, walk back across the court, and knock on the door again before exiting. Hence, participants performed each activity of *knock*, *walk* and *coffee* twice per trial over ten different trials, providing a total of 600 activity traces. To evaluate sensor accuracy in event detection, we present Receiver Operating Characteristic (ROC) curves for each sensor used in this setup. The ROC curves plot the *true positive rate* (TP_{rate}) against *false positive rate* (FP_{rate}) and depict the ability of the different sensor modalities to detect events at varying threshold levels of signal amplitude.

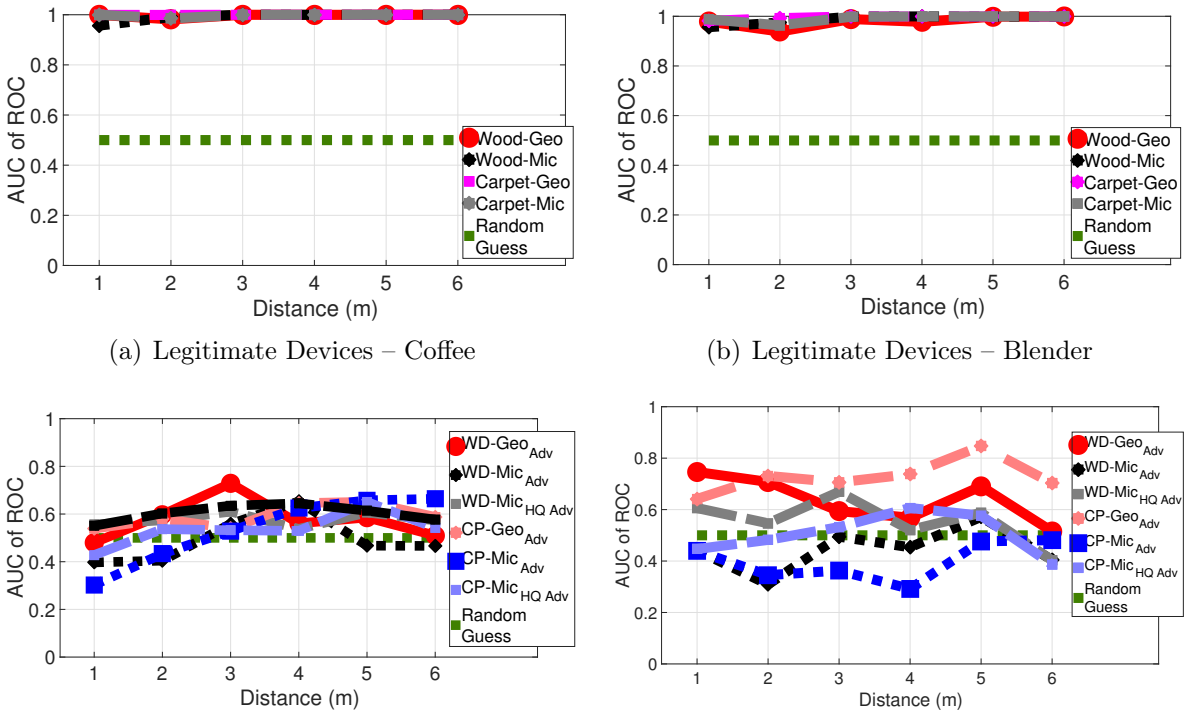
Figure 4.9 depicts the resulting ROCs by event type. For each event, we depict the ROC of only those sensors whose modalities would allow them to possibly detect it. For example, a motion detector cannot detect a *coffee* event, and hence is ommitted from the *coffee* ROC. We find that all legitimate sensors have a high signal detection accuracy as most Thr_{lower} yield a high TP_{rate} with relatively low FP_{rate}. For example, *knock* ROC depicts good detection abilities for the inside geophone, microphone, and accelerometer, yielding large area under the curve (AUC), while the motion detector and power meter do not produce any signal

for this event as expected (hence not shown). On the other hand, ROC curves for the $\mathcal{M}$s show relatively poor detection ability. We note that while all three events indicate that the higher quality attacker accelerometer and microphone generally perform better than their lower-quality counterparts, they are nevertheless unable to generate high TP_{rate} without generating equally high FP_{rate}. At best, their curves follow a random guess trend. Some of the ROCs, especially for the attacker, appear to be increasing in a piecewise step fashion rather than a smooth concave trend. This is due to the nature of ambient noise in the system. As the signal detection threshold is lowered, noise is detected as true positive until the threshold is lowered enough such that other (lower) ambient noise is detected as false positives.

4.5.2.2 Effect of Floor Types and Distances

We next study the effect of the floor type on the detection accuracy of $\mathcal{LD}$s vs. $\mathcal{M}$s. We vary the floor type between wood and carpet (most common variations found in homes) as depicted in Figure 4.8(b). For each floor type, we trigger two events sufficiently spaced apart with no overlap in signal detection: a coffee maker brewing (the same machine used from Section 4.5.2.1) and a blender (Cuisinart SPB-650 [43]) grinding. Since the accelerometer and motion detector cannot detect either, and the floor type does not affect the power meter, we study the sensing accuracy of the legitimate and attacker geophones and microphones. For each event type, the distance between the attacker/legitimate nodes and the event source (coffee maker/blender) is varied from 1-6m.

We show the resulting area under the ROC curve (AUC) for each sensor in Figure 4.10. Since the ambient noise inside the room is low (as is typical in homes), the legitimate geophone and microphone detect both the coffee and blender events with high accuracy for both floor types and across distances. The latter occurs due to the high signal to noise ratio inside the room even at longer distances from event source. On the other hand, the attacker's AUC fluctuates around 50% for carpet and wood alike across all distances for coffee events. Essentially, the attacker outside is contending with fluctuating noise levels due to the noisy surrounding, and is unable to detect these signals with accuracy any better than a random guess. For the blender event, the attacker geophone does show a slightly higher AUC, indicating better than random guess. This is as expected with the consistently higher sound and vibration caused by the blender as compared to the coffee machine. However, the attacker's AUC for blender, even for the geophone, barely exceeds 80% at best, and is significantly lower than the legitimate node's AUC.



(a) Legitimate Devices – Coffee

(b) Legitimate Devices – Blender

(c) Attacker Devices – Coffee

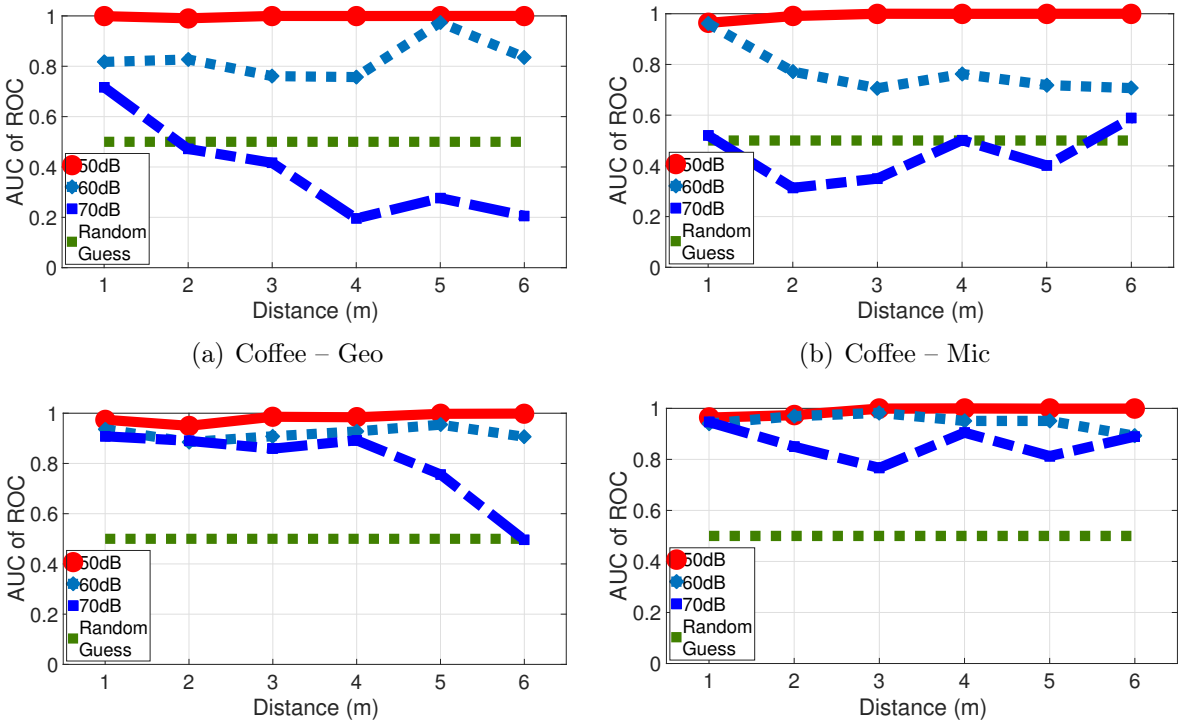
(d) Attacker Devices – Blender

Figure 4.10: As the distance between sensor devices and event source varies from 1-6m, the $\mathcal{LD}$s are consistently able to detect the event (with high AUC) while the $\mathcal{M}$s have a detection rate fluctuating around a random guess for carpet and wood alike. Since the blender is significantly louder with higher vibration than coffee brewing, the attacker's AUC is correspondingly higher.

4.5.2.3 Effect of Background Noise and Distances

While our analysis in Sections 4.5.2.1 and 4.5.2.2 show that $\mathcal{LD}$s consistently have high detection accuracy, the prevailing ambient noise inside the court was indeed low. We now study the degradation in event detection accuracy for the legitimate sensors with increasing background noise. Hence the background noise is varied between 50, 60 and 70 dB.

We show the resulting AUC for the legitimate microphone and geophone inside the room across distances of 1m to 6m from the event source in Figure 4.11. We see a clear trend of decreasing AUC across noise levels for all sensor types. As the ambient noise floor rises, the signal to noise ratio for the events degrades, incurring higher false positives for a given threshold of signal amplitude. At 50dB both the geophone and microphone are able to detect the coffee and blender with high AUC, with hardly any decline in detection rate from increasing distances to source. At 60 dB, the geophone's AUC for coffee is decreased compared to 50 dB, but remains mostly stable. The microphone, however, exhibits significantly degraded



(a) Coffee – Geo

(b) Coffee – Mic

(c) Blender – Geo

(d) Blender – Mic

Figure 4.11: For events *coffee* and *blender* alike, increasing noise levels result in poorer detection accuracy even for devices inside, as expected. Since the coffee machine has a significantly weaker signal than the blender, the degradation in detection accuracy is steeper for *coffee event* as the distance from source and noise level increases.

performance as the distance from the coffee maker increases at 60dB. As seen from previous analysis, the inherently higher sound and vibration generated by the blender results in the both sensors continuing to perceive it with high accuracy. At 70 dB, the signal to noise ratio for the coffee event degrades enough at higher distances to make its detection effectively a random guess for both nodes. Even for the blender, we see the geophone's AUC start to suffer at higher distances. Most home environments where *Perceptio* is suitable might have instances of high background noise (e.g. music playing loudly for a few minutes), during which sensors inside might not be able to fingerprint successfully. But as long as the environment exhibits ambient noise levels below 70dB for the most part, the sensors are able to detect events successfully for fingerprint extraction.

4.5.3 Key Establishment

4.5.3.1 Fingerprint Similarity between Legitimate Devices

While we demonstrated generally high event detection accuracy of legitimate devices, $\mathcal{LD}$s, under prevailing conditions inside the squash court in Section 4.5.2, this may not directly translate to satisfactory key establishment. This could be due to occasional detection errors, clustering errors, and relative temporal offsets in event detection between different sensor modalities. Hence, we evaluate our protocol in an end-to-end manner to demonstrate *Perceptio*'s ability to establish shared keys between $\mathcal{LD}$s (with heterogeneous modality) located within the physical boundary. To do so, we use real-world data to execute the *Perceptio* protocol and evaluate the fingerprint similarities F_{sim} between device pairs. Specifically, we first generate a data stream of three thousand events – consisting of *knocking*, *walking*, *coffee*, and *ambient noise* – by randomly drawing samples from the data set described in Section 4.5.2.1. Upon executing the protocol, we compute F_{sim} for all seven feasible sensor pairs across $\mathcal{LD}$s, as depicted in Figure 4.12 (note that there are ten sensing modality-pairs possible, but $\{acc, mot\}$, $\{acc, pow\}$ and $\{mot, pow\}$ are omitted as none of the tested events can be sensed in common by these pairs). We illustrate two interpretations of the fingerprint similarity for each sensor pair. First, we depict the overall fingerprint similarity across all fingerprint comparisons. The large standard deviation in this first set of bars reflects the variation across fingerprints that will be used and those that will be discarded due to low similarity. Second, we depict the average fingerprint similarity F_{sim} for only those fingerprints that are not discarded (i.e., those with similarity above the threshold). These are the fingerprints that actually contribute to secure key establishment and confidence. As depicted in Figure 4.12, all the sensor pairs that perceive at least one common event have high F_{sim} after the thresholding.

4.5.3.2 Confidence Score

Another important aspect of *Perceptio* is its *Key Strengthening Process*, which takes advantage of incremental growth in the confidence score (*ConfScore*) upon a successful iteration of key establishment protocol. Figure 4.13 depicts *ConfScore* of sensor pairs over time. As in the previous discussion, we depict the sensor pairs that perceive at least one event in common. The notion of time is depicted as the number of events arrivals in this figure, as more events arrive with more time (detailed modeling of event inter-arrival times and resulting time for

entropy extraction is presented in Appendix 4.5.5). From this figure, we have two important takeaways. First, sensor pairs that detect more events reliably and/or frequently in common exhibit a steeper increase in confidence. For example, $\{geo, mic\}$ pair perceives three events in common – *knock*, *walk*, and *coffee* – while $\{acc, mic\}$ perceives only the *knock* event in common. Hence we see that as more events arrive, *ConfScore* of $\{geo, mic\}$ pair increases faster than that of $\{acc, mic\}$. The pairs that do not reliably or frequently perceive a common event, such as $\{geo, mot\}$ have much slower increase in *ConfScore*. Second, it is important to note that *ConfScore* never decreases over time. Upon fingerprint mismatches (which contributes to lowered average F_{sim} in the first bar graphs of each sensor pairs depicted in Figure 4.12), the *ConfScore* levels off at the current state until the next successful fingerprint matching occurs. This means that any fingerprint mismatches – due to detection and/or clustering errors – *do not degrade* the key establishment process, but simply takes longer.

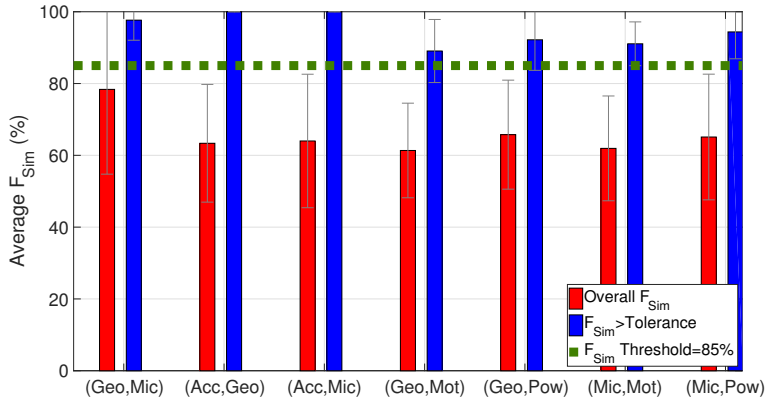


Figure 4.12: We verify that $\mathcal{LD}$s that sense common events are indeed able to pair with high fingerprint similarity. Occasional inaccuracies in event clustering and temporal offsets in event detection cause the average fingerprint similarity between modality-pairs to be around 65% with a high variance. However, even at 85% similarity threshold for successful pairing, all sensor modalities manage to establish keys within a few successful tries, with low variance.

4.5.3.3 Fingerprint Similarity between Attacker and Legitimate Devices

It is evident from the attacker's event detection ROC studied in Figures 4.9(d) 4.9(e) 4.9(f) that the $\mathcal{M}$s can hardly perform better than random guess. Further, given that requisite clustering also incurs some errors, it is expected that the likelihood of an $\mathcal{M}$ achieving a high F_{sim} with an $\mathcal{LD}$ can be no better than a random guess. We nevertheless evaluate this by further granting two unfair advantages in favor of the attacker. First, we assume that

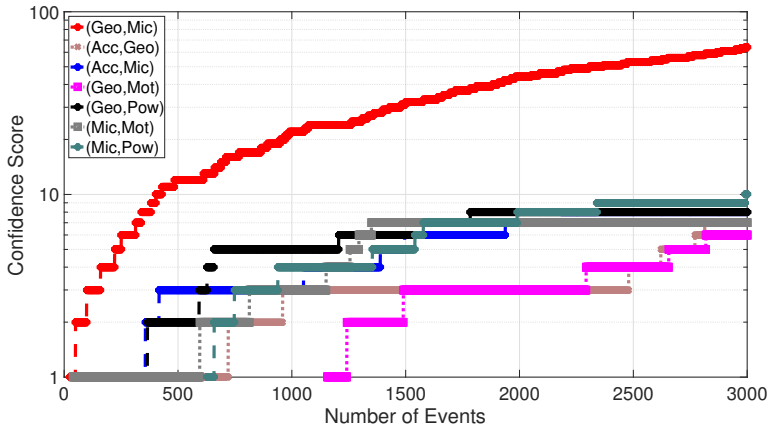


Figure 4.13: We study the *key strengthening* process by observing the increase in confidence score for each established legitimate sensor pairing as the number of encountered events in the environment increases. Modalities such as geophone and microphone that are able to simultaneously sense most of the occurring events exhibit a much steeper increase in confidence scores as compared to pairings such as $\{geo,\ mot\}$ that sense relatively fewer events in common.

the $\mathcal{M}$s are capable of yielding less errors in event detection. There are two types of errors in event detection – *insertion* and *deletion errors*, each represented by FP_{rate} and TP_{rate} respectively. We only considering errors due to deletion, and assume that the $\mathcal{M}$s do not yield any insertion errors – i.e., yielding high TP_{rate} with no FP_{rate}. From the ROC curves aforementioned in Section 4.5.2, we choose the best possible TP_{rate} for each attacker sensor that corresponds to $FP_{rate} = 50\%$, but replace the FP_{rate} to 0%. Second, we assume that the attacker has 100% clustering accuracy.

While these are unrealistic advantages, we evaluate F_{sim} with such assumptions to account for the chance possibility that the attacker may detect events at a higher accuracy or have access to better clustering methods. Hence, the two advantages provide an optimistic scenario for the attacker.We evaluate fingerprint similarities between $\mathcal{M}$s and $\mathcal{LD}$s with a simulated stream of events by exhaustively searching for best matching fingerprints. Figure 4.14 depicts the reported values, with a maximum value of 70% between the attacker and legitimate geophones. Recall from Figure 4.12 that we draw the requisite similarity threshold at 85%. Hence the attacker's best case F_{sim}, even with the unfair advantages, are sufficiently below the tolerance level, demonstrating that *Perceptio* succeeds in thwarting the attack.

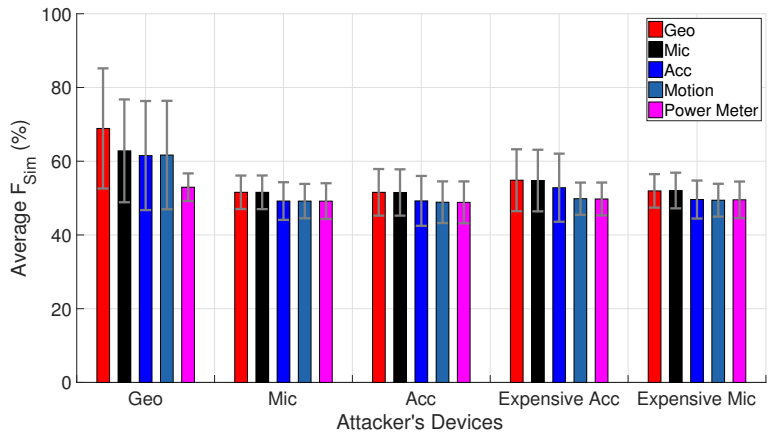


Figure 4.14: We present a simulated study of F_{sim} for $\mathcal{M}$s attempting to pair with $\mathcal{LD}$s. Even with overestimated capabilities of the attacker, average of all F_{sim} is only at 55%, bar the expensive geophone (around 70%), but nevertheless sufficiently below the *tolerance* line of 85% set in Figure 4.12.

4.5.4 Security Analysis

We now present the analysis of *Perceptio*'s cryptographic protocol, namely presenting how an attacker would try to launch attacks to compromise the shared secret. Specifically, the attacker's goal is to acquire k_i generated by A in Figure 4.5 Step 2. We analyze two types of attacks that an attacker may launch to achieve the aforementioned goal – *(1) bruteforcing* and *(2) eavesdropping* attacks.

(1) Bruteforcing attack. The attacker first tries to directly bruteforce the key, k_i by attempting to perform dictionary attack on the hash, $H(k_i)$, which is transmitted together with C_{A_i} in Figure 4.5 Step 3. As long as the length of the cryptographic hash function ($H(\cdot)$), $l_{H(\cdot)}$, is longer than l_{NIST} bits, bruteforce attack is computationally infeasible (i.e., $l_{H(\cdot)} \geq l_{NIST}$ bits). We leverage the state-of-the-art secure cryptographic hash function such as SHA-3 [33], which is well above l_{NIST} bits. We define $l_{NIST} = 112bits$, as recommended by NIST [31].

(2) Eavesdropping attack. A more sophisticated attacker pretends to be a legitimate device placed within the physical boundary by trying to open the commitment. The attacker launches an *eavesdropping attack* to try to capture some of the events by placing his/her devices just outside of the physical boundary. Hence, these devices may capture some of the signals, depending the transmission media as well as the amplitude of the original signal. Hence, rather than performing a bruteforce attack with no known information, the attacker has more information at guessing the fingerprint, which can be decoded with $DEC(\cdot)$, which

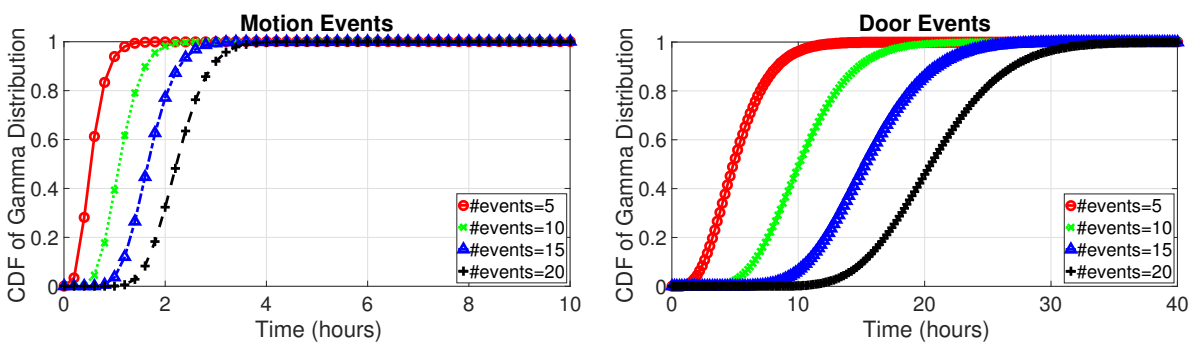
in turn leads to less amount of computations to acquire k_i.

We denote l_{eaves} as the number of bits of the fingerprint that the attacker knows as a result of the eavesdropping attack. Hence, we denote l_{bf} as the number of bits the attacker needs to bruteforce in order to successfully know l_{tol} bits in order to succeed in the attack, such that $l_{bf} = l_{tol} - l_{eaves}$. Hence, the attacker's success probability is $P(Adv) = 1$ with computational complexity, Cpx, as following:

$$\begin{aligned} Cpx &= p2^{l_{bf}}(Ops + \ominus + DEC(\cdot) + H(\cdot) + V_{H(\cdot)}) \\ &\approx O(2^{l_{bf}}) \end{aligned}$$

where p is the number of Fs and $V_{H(\cdot)}$ is hash verification. Cpx is computationally infeasible if $l_{bf} \geq l_{NIST}$. Hence the gain from eavesdropping, l_{eaves} should be bounded by $l_{eaves} = l_{tol} - l_{NIST}$.

4.5.5 Evaluating Entropy Extraction



(a) Motion (b) Door

Figure 4.15: Cumulative probability distribution of *motion* and *door opening* events modeled after real world smart home data collected for two months

We now evaluate the required time to extract l_F (i.e., length of fingerprint) to ensure sufficient entropy (e.g., 128 bits). F is created by concatenating the time intervals of consecutive events of same cluster type (e.g., series of knocking events).

4.5.5.1 Modeling the Arrival Time

We follow the traditional approaches of modeling event arrivals as a Poisson process [71, 91]. We define S_n as the waiting time until the nth event, assuming that n events yields l_F bits of

fingerprint. We define T_i as the sequence of inter-arrival times for $i = 1, 2, ...$, which can also be described as i.i.d. exponential random variables. Furthermore, the probability density function of S_n has a gamma distribution with average arrival rate λ, number of events n, and time t as depicted in Equation 4.3.

$$S_n = \sum_{i=1}^{n} T_i, \quad n \geq 1, \qquad f_{S_n}(t) = \lambda e^{-\lambda t} \frac{(\lambda t)^{n-1}}{(n-1)!}. \tag{4.3}$$

The corresponding expected time of nth event, $E(S_n)$, is depicted in Equation 4.4. We also define bit rate which is the effective rate of the generating the l_F fingerprint bits in a time duration of $E(S_n)$, capturing the effective rate of generating useful fingerprint bits. The bit rate is modulated by a correction factor, ρ, which is proportional to the detection rate of the events. We note that the units of the bit rate can be measured in bits per second, but in many practical scenarios it may be more meaningful to express this value in bits per hour.

$$E(S_n) = \frac{n}{\lambda}, \qquad BitRate = \frac{l_F\,\rho}{E(S_n)} = \frac{\lambda\, l_F\,\rho}{n} \tag{4.4}$$

4.5.5.2 Evaluation Using a Real-world Smart Home Dataset

To ensure the practicality of our analysis, we analyze a real-world smart home data set, publicly available from CASAS online repository [57]. We analyze two sets of sensor data collected for two months (i.e., over 1450 hours of data): a motion detector used to monitor movement in the home, and a door sensor to monitor door open/close activities. Specifically, we extract mean arrival rate of the two events, λ_{motion} and λ_{door}, to be 8.85 events/hour and 0.96 events/hour, respectively. We note that the average was computed from the users' daily activities only. This reflects the practical use case of *Perceptio*, as the system will not extract much entropy at night due to stagnant event occurrences. Using these values, we plot a cumulative probability density function (CDF) and vary n and t. Figure 4.15 (a) and (b) depict the CDF of the two types of events, respectively. The results are intuitive as the plots demonstrate that for more number of n events, the longer t is required to reach a high probability. Furthermore the two figures of motion and door events depict clear contrast, as the door events require much longer time to reach a high probability. We note that this analysis is an optimistic approach as we assume perfect detection accuracy (i.e., $\rho = 1$) for simplicity of the analysis.

For example, assume that it takes 20 events to yield $l_F = 128$ *bits* of the fingerprint,

then using (Equation 4.4), $n = 20$ events arrive in about 2.3 hours for motion events, as opposed to 20.8 hours for door events. Hence, the corresponding bit rate for the two events are $BitRate_{motion}$ of 56.6 bits/hour and $BitRate_{door}$ of 6.1 bits/hour. We note that the bit rate would potentially increase if there were more occupants in the house as opposed to a single resident case from this data set. (For example, the average number of occupants in a home in the United States is 3.14 persons [36]).

4.6 Related Work

We present related work in automatic trust establishment enabling the IoT devices to leverage contextual information. Miettinen et al. propose recurring authentication when pairing IoT devices at home by leveraging contextual information (i.e., light and sound) [99]. Devices co-located at one household would experience similar context as opposed to devices in a neighbor's house. This scheme increments authenticity score over time to derive a shared key among the devices with certain confidence. Schurmann et al. propose a similar idea where they leverage short audio as contextual information to be used for pairing [117].

There are work in non-residential settings that make use of contextual information to achieve security guarantees. Han et al. proposes leveraging context from road characteristics to prove co-presence of trucks in platoon [68]. Rostami et al. propose Heart-to-Heart, a key agreement scheme between an implanted heart with its remote programmer [114]. The two devices establish a shared key by extracting entropy bits from measuring the patient's heart beat.

While these work make use of contextual information to verify co-presence, all of their approaches rely on leveraging same sensing modality across devices. These approaches are promising first steps but they focus on leveraging identical sensor pairs such as microphones, accelerometers, microphones, and other sensors using direct signal analysis. Unlike the traditional line of work that make use of homogeneous sensing modalities, *Perceptio* addresses a difficult but interesting question of how to enable differing (i.e., heterogeneous) sensor modalities to capture the same context information.

4.7 Discussion

We now discuss practical considerations when deploying *Perceptio* in smart homes.

Simultaneous Events. While we present experiment evaluations with a single event per

time period and background noise, this may not always be true in real life, as multiple events may occur simultaneously (e.g., coffee making while walking). In such cases, we have seen that the concurrent events will produce an overlapping signal and either be clustered as a separate event type or mismatch errors will occur leading to a longer time to reach the confidence threshold. To test our hypothesis, we conducted a preliminary experiment with two events – *coffee* making and generating *footsteps* (walking in place) – occurring simultaneously, while the sensors were located 1m away from the event sources. We then kept the locations of sensors and the coffee machine static, while varying the stepping positions from 1-6m. Figure 4.16 depicts an example plot of signals captured at 1-4m distances between the simultaneous events. At 1m distance, the signals differ significantly from those generated by the coffee machine and footsteps in isolation, while at 4m distance, the signal characteristics are closer to those of a coffee machine in isolation. We see that many overlapping signals will lead to new event clusters of their own, rather than with existing event types.

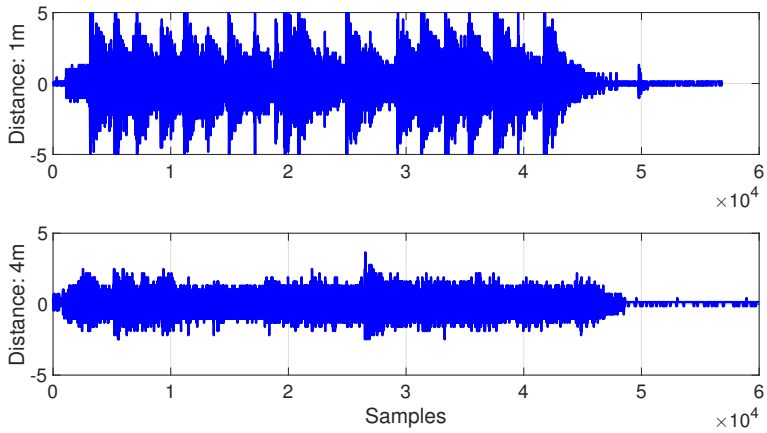


Figure 4.16: When events *coffee* and *footsteps* occur simultaneously, the combined signals are distorted significantly enough to possibly cluster into a new event type of its own. However, the magnitude of this distortion also depends on distance between event sources.

Ad-hoc networking. *Perceptio* provides a novel solution to secure ad-hoc connectivity among IoT devices, *without* the need for a trusted home gateway. Many applications may benefit from such ad-hoc networks due to reduced communication and computational overhead, as it no longer requires going through a central gateway or cloud. In fact, there is a push in the industry to shift from star to mesh topologies, as seen by industry activities such as Thread [65].

Resourceful attackers. Through our evaluation, we demonstrated the difficulty of the attacker succeeding in Shamming–Eavesdropping attack due to the need to consistently

detect events inside the home. However, if an attacker launches Shamming–Signal Injection attack by creating and injecting events from outside that are consistent and loud enough to be sensed from legitimate devices inside the house, the attacker may succeed in fooling the legitimate device to pair. However, due to the same attenuation factor that protects inside events from the external attacker, it would be difficult for inside devices to consistently detect outside events unless they are *extremely loud*, otherwise the fingerprints would not match. To make this attack harder, *Perceptio*'s *Key Strengthening Process* requires multiple iterations that take enough time that such injections would be easily noticed by human users, making the attack extremely risky and likely impractical. Furthermore, our threat model also defines Shamming–Sensor Spoofing attack by injecting spoofing signals to sensors of legitimate devices similar to prior work [120, 121, 124, 128]. While such attacks may still be possible, *Perceptio* would cluster injected signals into another event type. Hence, the attacker would only slow down the *Key Strengthening Process*. Furthermore, such injection attacks require a high amplitude signal to be exerted to the sensor, which is rather difficult in our setting as signals attenuate significantly through the wall as our experiments have shown.

Devices located in different rooms. *Perceptio* is potentially unable to establish trust between valid devices located in different rooms of a smart home. A possible remedy is to introduce a *bridging device* in each room to facilitate cross-room connections. A bridging device would be like any other IoT device, but with the additional functionality for human-in-the-loop pairing. For example, two infrared- and NFC-enabled motion detectors in different rooms may be first manually paired by the user (e.g., via NFC tagging with a smartphone) and then deployed to each room. Devices in each room can leverage the Transitivity-of-Trust (ToT) protocol (Section 4.4.1) via the bridging devices to pair with devices in other rooms. Manually bootstrapping bridging devices is reasonable because there are only as many bridging devices as rooms in the home. This is analogous to distributed WiFi systems that use multiple APs to provide or enhance connectivity through a large home [53, 62, 112].

Calibration. *Perceptio* depends on sensor calibration and determination with respect to appropriate threshold values presented in Section 4.4.2.1. Thresholding is important to helps distinguish signals from noise and is thus critical with respect to factors such as sensor placement, sampling rate, and events in the environment. Hence, in practice, *Perceptio* would require a calibration phase by allowing the IoT devices to perform local sensor calibrations for a given amount of time prior to starting the *Perceptio* protocol. Device manufacturers could also provide course-grained pre-calibrated settings.

Public and Shared Spaces. *Perceptio* is based on the assumption that physical

boundaries draw natural barriers between the legitimate devices and the attacker's device outside, which may not hold for public spaces such as public libraries or shopping malls. However, with further work on fine tuning thresholding parameters, *Perceptio* can be extended from single family housing to other multi-tenant private office buildings with existing access control policies.

Frequency of activity vs. Pairing time. The pairing time between devices is directly proportional to the frequency of activities in the house. However, there may be households with less family members and thereby decreased sensor activity, leading to undesirably long pairing times. In such cases, users may introduce a *signal injecting device* for faster convergence. This solution, however, trades procurement cost and usability for speed.

4.8 Chapter Summary

We propose *Perceptio* for autonomous, secure pairing of IoT devices using context information from embedded sensors. The novelty of *Perceptio* stems from its ability to address the difficult challenge of context-based pairing across devices equipped with different types of sensors. *Perceptio* achieves this goal by abstracting sensor measurements and using timing information as an invariant property to generate context fingerprints as a source of shared entropy for cryptographic key agreement. We demonstrate through proof-of-concept experiments that *Perceptio* is able to securely pair heterogeneous sensing devices co-located within the same physical boundary, while rejecting potential attacker devices placed outside.

Chapter 5

Least Constrained: Truck Platooning

We now present an application scenario with the least constraint among the three examples we present in this chapter. We present a truck platooning scenario, where there is no physical boundary that exists to keep the unintended devices out. Rather, the trucks may be traveling openly, with the possibility of attackers driving along side the trucks.

5.1 Problem Definition

Vehicle platooning is a method of having a group of vehicles drive in a single file to follow the preceding vehicle and ultimately the leading vehicle, which is gaining large traction today. Platooning is getting significant attention in the commercial trucking industry [12, 15, 17, 87, 138] as it provides benefits of increase in fuel efficiency (and reduction in CO_2 emissions), driving safety, road efficiency , and driver convenience [16]. The consumer vehicle industry is also preparing to incorporate personal vehicles into platooning to take advantage of the aforementioned benefits [18, 113].

Because platooning leverages wireless communications to transmit control messages such as accelerating, braking, and steering information from the participating cars, securing the communication is extremely crucial as attacks may result in life-threatening collisions, damage to high-value vehicles and cargo, and loss of business. Specifically, vehicle platooning uses Dedicated Short-Range Communications (DSRC) and Wireless Access in Vehicular Environments (WAVE) as de facto standards for vehicle-to-vehicle (V2V) communications [78, 90]. The current DSRC/WAVE model assumes Public Key Infrastructure (PKI) that authenticates each vehicle's public key by leveraging certificates signed by a trusted third party, such as a Certificate Authority (CA). Unfortunately, this model is susceptible to impersonation

attacks such as masquerading or sybil attacks (impersonating as non-existing or "ghost vehicles") [34, 89].

The root cause of the aforementioned problems is due to the fact that the vehicles have no way of binding their digital certificates with their physical identities. This is on the contrary to the analogy of web server authentication in TLS/SSL. The certificates of the server contains the identifier (i.e., URL), which can be compared to the URL that the user has visited via the web browser, hence naturally *binding* the "physical identity" with the identity included in the certificate. Even though the certificate of the vehicles contain their identifiers, other cars in the platoon have no way of ***verifying*** the physical identity of the certificate. This is exemplified in Figure 5.1, where *Cars A* and *B* are vehicles in am existing platoon. *Car C* is a vehicle that wishes to join the platoon, and *Car M* is an attacker's car driving in an adjacent lane. In this example, *Cars A* and *B* receives *Car C*'s certificate, but is unsure if the certificate is actually from *Car C* or *Car M* in the adjacent lane.

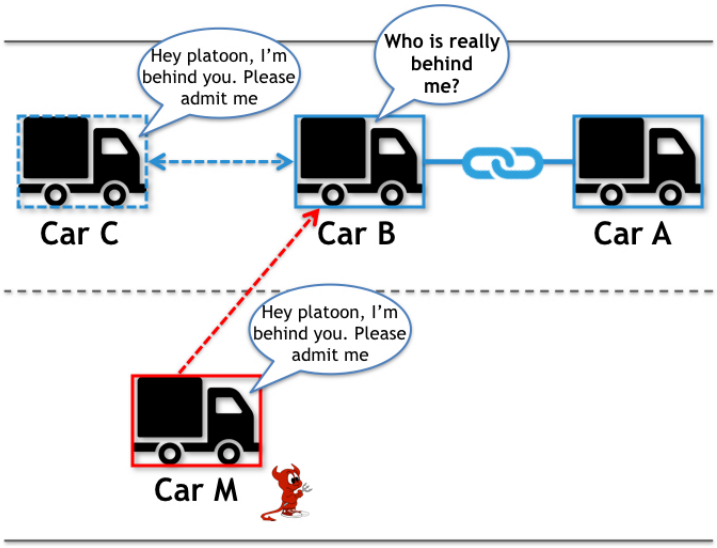


Figure 5.1: Overview diagram depicting vulnerabilities of platooning systems to impersonation attacks.

To address the above problem, we present *Convoy*, an autonomous authentication and verification scheme of platooning vehicles. *Convoy* performs the verification by binding the certificate with the physical context. *Convoy* is based on the findings that vehicles wishing to form a platoon can prove to each other that they are indeed traveling together using context information captured from their sensor data. This is possible because *Convoy* requires a vehicle wishing to join the platoon (*Car C*) to follow a rear vehicle of the platoon (*Car B*) for

a period of time resorting to automated technologies such as Adaptive Cruise Control (ACC), which enables a vehicle to autonomously follow the front car with constant headway and keeps the vehicle within the traveling lane (ACC is prevalent in many vehicles today) [11, 133]. Consequently, unique *road conditions* per lane (e.g., bumps, cracks, potholes, etc.) cause similar vibrations between potential vehicles of a platoon as they travel on the same lane in a single file, as opposed to a car traveling on an adjacent lane (Cars A, B and C as opposed to Car M). Furthermore, *traffic conditions* cause vehicles traveling in a single file to experience similar acceleration, deceleration, braking, and steering. Hence *Convoy* leverages these conditions as sources of entropy to establish a symmetric cryptographic key between a pair of vehicles, naturally binding the physical context to the symmetric key. Subsequently, the symmetric key is used to authenticate the certificates to delegate the bindings of physical context to the certificates.

Yet, challenges remain to achieve the aforementioned goals. First, different cars wishing to join as a platoon (e.g., Cars A, B, and C) would experience similar but different context, leading to numerically unequal signals. In order to compensate for subtle differences in the signals between the cars, *Convoy* makes use of an emerging cryptographic primitive called *Fuzzy Commitment* [76, 99] that relies on error-correcting codes to establish a shared symmetric key from similar-but-unequal signals capturing a common context.

Second, comparison of contextual information increases room for error because similar context may exist for attackers at times (e.g., two lanes may have similarities at times). Hence, *Convoy* requires the vehicles to repeat the protocol for multiple iterations over time (and increment a confidence score upon a successful termination of each iteration), thereby increasing the probability of vehicles traveling together to experience more similar context, while vehicles traveling in another lane would not. Hence, *Convoy* thwarts attackers driving on an adjacent lane to the platoon, attempting to claim a membership in the platoon or impersonating an existing member in the platoon.

5.2 System Models

In this section, we present our models and assumptions for the attacker and vehicle platoons.

5.2.1 Adversary Model

We consider a ***Platooning Ghost Attack***, where the attacker's goal is to impersonate a non-existing "ghost" vehicle in the platoon. By pretending to be in the platoon formation, and hence gaining admittance to the platoon, the attacker gains knowledge of the control commands (i.e., acceleration, braking, and steering information) from the preceding vehicles relative to the position of the ghost vehicle. The attacker further has control over transmission of the control messages to its succeeding vehicles. Hence, the attacker effectively controls certain aspects of the platoon. The attacker is now capable of launching a variety of attacks as a *platoon insider*, including man-in-the-middle, denial-of-service, and collision induction attacks. More specifically, it may send malicious control messages to its succeeding vehicles to cause it to crash into the rest of the platoon in front. It may prevent admissions of newer members of the platoon, or cause existing succeeding vehicles to brake away from the rest of the platoon.

5.2.2 Platoon Model

Platoons are typically set up with a manually-driven lead vehicle with semi-autonomous followers [138]. In our work, assume that a candidate vehicle is only admitted to the platoon after the rear-most platoon vehicle validates the position and identity of the candidate. We suppose that the candidate will initially follow the platoon using Adaptive Cruise Control (ACC) [133] without explicit coordination, until it can be verified and admitted to the platoon. Once admitted, members are declared to be trustworthy and thereby earn the benefits of efficiency and safety offered by platooning [16]. To enable the coordinated acceleration among vehicles, vehicle-to-vehilcle (V2V) communication is employed. As platoons travel amongst other traffic, it is critical for them to communicate securely. We thus assume that all control messages (e.g., acceleration, brake, and steering messages) are encrypted with a group symmetric key known only to the platoon members, though we do not address group key management in this work.

5.3 Capturing Contextual Cues

The goal of *Convoy* is to enable vehicles trying to establish a secure platoon to verify that their public keys indeed belongs to the vehicles driving in the platoon as opposed to an attacker's car driving in an adjacent lane. Hence, the protocol binds the physical context,

namely innate context experienced by the vehicles driving together in a platoon, to their public keys. Specifically, *Convoy* enables this binding by making use of the shared context as sources of entropy in establishing a shared symmetric key between a pair of cars. This symmetric key is used to verify the owner of the public keys by having the pair of vehicles each compute a MAC over each car's public key.

Convoy leverages the following innate context for a platoon to leverage as sources of entropy to the key agreement protocol – unique *road* and *traffic conditions*. First, each lane has *varying road conditions* that are hard-to-guess by attackers. The road conditions not only differ at different segments of the road, but also across different lanes. This is often due to many factors such as patches, bumps, cracks, pot-holes, etc. Second, traffic conditions are inherently random as they vary when different vehicles on the road travel together, causing the platoon to accelerate, brake, and steer differently. Hence, the vehicles traveling on the same lane within a platoon will experience similar road and traffic conditions, while an attacker traveling in an adjacent lane will not. We note that the control messages (e.g., acceleration, brake, and steer messages) are encrypted to be shared only within the members of a platoon to thwart attackers from mimicking the traffic conditions. However, a new vehicle wishing to join a platoon (*Car C*) follows the read vehicle of the platoon (*Car B*) resorting to ACC so that C's traffic information would be similar to that of B. This is depicted in Figure 5.1, where Vehicles A and B already travel as an existing platoon and Vehicle C is trying to join the platoon, while an attacker's vehicle, M, is traveling in an adjacent lane.

Convoy leverages the aforementioned findings to extract context fingerprints to have a newly joining vehicle (C) to prove to the vehicles in a platoon (A and B) that they are traveling together. Each vehicle running *Convoy* protocol makes use of a multi-axis accelerometer to capture the context. Specifically, the road condition is captured by an axis perpendicular to the road (Acc_{Road}), while the traffic condition is captured by the axis parallel to the lane (Acc_{Traf}). The captured context is then extracted to fingerprints, which will be used in the protocol shown later in this chapter to prove to each other that they are in fact driving together in a platoon.

5.4 Design and Implementation

This section presents the design of *Convoy* protocols and the corresponding implementation details.

5.4.1 Protocol Overview

From the example illustrated in Figure 5.1, in order for *Car C* to prove to *Car B* that it is traveling close behind the platoon, it leverages a *fuzzy commitment scheme*. This scheme translates sensor measurements, represented by an extracted fingerprint F, and a secret K into a commitment and decommitment (or opening) pair (μ, o). This is analogous to one-time pad encryption, where F is used as an encryption key, and K is used as the plaintext to be encrypted. μ can only be opened if one has a fingerprint, $\hat{F}$ that is within a few bit errors of F. The fingerprints F_B and F_C extracted by B and C traveling on the same lane would ideally be within a small margin of error, while F_M extracted by M on a different lane would be more error-prone. By applying an error-correcting code operation, vehicle pairs can verify fingerprint similarity, resulting in a shared key. This can be repeated to build confidence over time, ultimately yielding a key with sufficient entropy and corresponding platoon admission. In this work, we rely on a fuzzy commitment scheme similar to that of previous work [76, 99, 117].

5.4.2 Protocol Details

Convoy protocol consists of five phases – (1) Initialization, (2) Key Agreement, (3) Key Confirmation, (4) Public Key Verification, and (5) Confidence Score Check phases. We describe each phase in detail with the platoon example depicted in Figure 5.1. The protocol is summarized in Figure 5.2.

Initialization Phase. To start the initialization phase of *Convoy*, the platoon leader A broadcasts a beacon message $Beacon_A$ containing current platoon member IDs, their (GPS) locations, and a timestamp. When platoon candidate C receives several beacons, it sends a request $JOIN\_RQST$ to join the platoon. Upon receiving the request, A sends a message $INIT\_VERIF$ to B (the trailing vehicle, in general) and C to initialize the verification process; A names C in this message and includes the measurement duration t_F. At this point, B and C commence the key agreement phase.

Key Agreement Phase. This phase is performed by the trailing platoon member (B in our example) and the candidate vehicle C. When A triggers the $INIT\_VERIF$ messages, B and C collect accelerometer measurements for a duration of t_F seconds. The vehicles then apply a fingerprint extraction function *extractF()*. B computes fingerprint as $F_B = extractF(Acc_B, t_F)$, where C does the same for its measurements Acc_C. We present the details of our fingerprint extraction algorithm in Section 5.4.3. Subsequently, B generates

***Convoy* PROTOCOL**

Phase 1: Initialization

1. $A \xrightarrow{bcast} All$: $Beacon_A = ID_A || ID_B || TS || GPS_A || GPS_B$
2. $C \rightarrow A$: $JOIN\_RQST$
3. $A \rightarrow B, C$: $INIT\_VERIF(ID_C, t_F)$

Phase 2: Key Agreement

4. B : Holds F_B, K_B, (μ_B, o_B)
   C : Holds F_C
5. $B \rightarrow C$: $\mu_B || H(K_B)$
6. C : $\hat{o}_B = Open(\mu_B)$
   : $\hat{K}_B = RS_{dec}(\hat{o}_B)$; $H(\hat{K}_B) \stackrel{?}{=} H(K_B)$
   : $\hat{K}_{BC} = KDF(\hat{K}_B)$

Phase 3: Key Confirmation

7. $B \leftrightarrow C$: Key confirmation messages for K_{BC}

Phase 4: Public Key Verification

8. $B \rightarrow C$: $m_{B_1} || M_{K_{BC}}(m_{B_1})$, where $m_{B_1} = K_B^+ || K_A^+$
9. $C \rightarrow B$: $m_C || M_{K_{BC}}(m_C)$, where $m_C = K_C^+$
10. $B \rightarrow A$: $m_{B_2} || M_{K_{AB}}(m_{B_2})$, where $m_{B_2} = K_C^+$

Phase 5: Confidence Score Check

11. B, C : Increment CS_{BC}; Check if $CS_{BC} > Thr$
    Repeat Steps 4 - 9 until check passes

Figure 5.2: *Convoy* protocol overview. Upon successful completion of this protocol, *Car C* is securely admitted to the platoon with existing members *Cars A and B*.

K_B using a key generation algorithm $KGen$ that outputs keys of length γ (e.g., 128 bits). B's commitment and opening pair (μ_B, o_B) is then computed as $o_B = RS_{enc}(K_B)$ and $\mu_B = F_B \ominus o_B$, where RS_{enc} and $\ominus$ denote Reed-Solomon (RS) encoding and subtraction in a finite field (analogous to an XOR operation), respectively. Finally, B sends μ_B and $H(K_B)$ to C. The hash is sent so that C can locally verify the opening of the commitment. Upon reception of μ_B, C first tries to open the commitment ($Open(\cdot)$) by inverting the operations using its fingerprints F_C in place of B's commitment, obtaining $\hat{o}_B \approx F_C \ominus \mu_B$. As long as $F_B \approx F_C$, the resulting $\hat{o}_B$ will also be similar to o_B. Applying RS decoding operation will yield a key $\hat{K}_B = RS_{dec}(\hat{o}_B)$ that will be equal to K_B if and only if the input fingerprints F_B and F_C are within the error-correction threshold t of the RS code, $\|F_B - F_C\|_1 \leq t$, where $\|\cdot\|_1$ is the Hamming distance (or ℓ_1 norm), counting the number of bit errors between F_B and F_C. Vehicle C then verifies that the acquired $\hat{K}_B$ value matches those computed by B by checking the original hash received from B as $H(\hat{K}_B) \stackrel{?}{=} H(K_B)$. Upon successful verification,

C computes a shared symmetric key, K_{BC} using a Key Derivation Function as $KDF(\hat{K}_B)$. B and C then continue to the key confirmation phase.

Key Confirmation Phase. C initiates the key confirmation phase by leveraging the newly computed K_{BC} to challenge B to verify the same key K_{BC} was derived by both parties. To construct the challenge β_1, C computes a Message Authentication Code (MAC) using K_{BC} over a random nonce n_C such that $\beta_1 = n_C||MAC_{K_{BC}}(n_C)$, and sends β_1 to B. Upon receiving the challenge, B similarly computes K_{BC} as $KDF(K_B)$ and verifies the received MAC using its version of K_{BC}. When this verification succeeds, B similarly creates its own challenge α with nonce n_B, such that $\alpha = (n_B||n_C)||MAC_{K_{BC}}(n_B||n_C)$ and sends α to C, who similarly verifies α. Upon successful verification, C transmits a final MAC β_2 over n_B received from B such that $\beta_2 = n_B||MAC_{K_{BC}}(n_B)$. At this point, both B and C have confirmed mutual agreement upon the symmetric key K_{BC}.

Public Key Verification Phase. With a confirmed symmetric key between the platoon trailer B and candidate vehicle C, the platoon provides verifiable public keys of all platoon members. Specifically, B computes a MAC over the public keys K_A^+ and K_B^+ and transmits the public keys and MAC values to C. C mirrors the process and transmits its public key and corresponding MAC to B. If desired, B can share this information internally within the platoon group, using the shared group key, in case B leaves the platoon before C completes the final phase.

Confidence Score Check Phase. After key confirmation and verification, B increments its (or the group's) confidence score CS_{BC} in candidate vehicle C. If CS_{BC} has surpassed a pre-defined threshold Thr, then C is admitted to the platoon and given access to the group key. Otherwise, C remains a candidate and must repeat the process from the key agreement phase until sufficient confidence is achieved. Use of the confidence score minimizes false positives and ensures that over time, B and C must be traveling together in the same lane.

5.4.3 Fingerprint Extraction Algorithm and Implementation

The *extractF()* function takes in a raw signal (i.e., Acc_{Road} or Acc_{Traf}) and the fingerprint capture time, t_F, to encode the signal to a bit stream, F, of length l_F. The main idea behind this extraction algorithm is to capture abrupt changes in the raw signals and encode them into *high bits* (i.e., bit value '1's), while encoding the rest of the signal as *low bits* (i.e., bit value '0's). The extraction is divided into the following phases: (1) Pre-processing, (2) Derivative, (3) Bit Translation. The phases are illustrated in Figure 5.3.

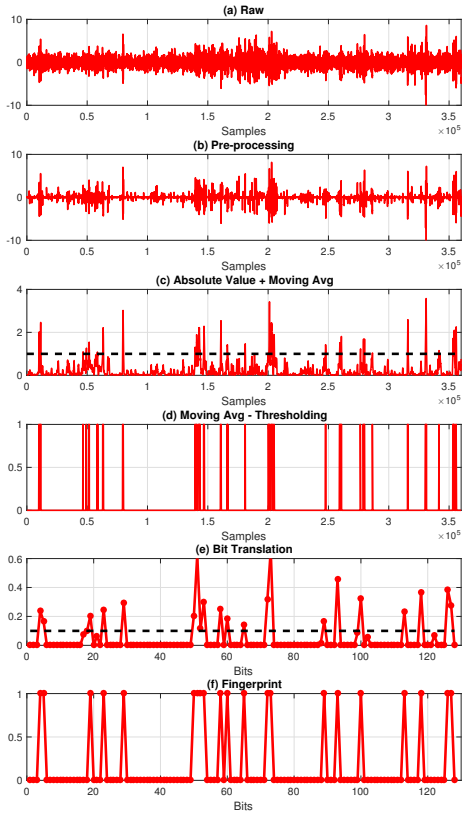


Figure 5.3: Fingerprint extraction depicting (a) raw data; (b) noise reduction phase; (c) absolute value and moving average; (d) binary signal after thresholding; (e) bit translation phase(total bit length is 128); and (f) extracted fingerprint.

Pre-processing. We pre-process raw sensor data to minimize noise and maximize detection of information (i.e., context from road and traffic conditions). Hence, first process raw sensor data to remove high frequency noise components in the raw data which would otherwise appear as noise in the following steps. We compute the absolute value of the entire signal to capture the magnitude of the samples independent of the sign. Next, with goals to remove high frequency noise in the data, we compute the average of samples as a sliding window, computing a moving average of the raw signal.

Derivative. Taking the pre-processed signal, we take the derivative of the signal to obtain, $S'[x]$. The derivative helps to detect abrupt changes in the smoothed signal. The derivative of the signal with respect to time rewards abrupt changes in the signal while penalizing slow stagnant changes. We note that this is computationally synonymous to applying a bandpass filter to capture the signal of interest, while implementation may be realized in different ways. This is depicted in Figure 5.3 (c). The dotted lines indicate the thresholding value, Thr_{Deriv},

to capture sudden changes. The resulting binary signal, $S'_{binary}[t]$ is computed as Equation 5.1 and illustrated in Figure 5.3 (d).

$$S'_{binary}[t] = \begin{cases} 1, & \text{if } S'[t] > Thr_{Deriv} \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

Bit Translation. $S'_{binary}[t]$ is then divided into l_F windows (e.g., 64 windows for 64 bits) of size $bitWnd$. For each window, we compute the summation of the values of $S'_{binary}[t]$. The resulting summation is depicted in Figure 5.3 (e). If this sum is greater than a threshold (depicted with the dotted line), Thr_{trans}, the encoded bit is 1, and 0 otherwise, for the fingerprint, F, as depicted by Equation 5.2.

$$F[n] = \begin{cases} 1, & \text{if } \sum_{t=0}^{bitWnd} S'_{binary}[t] > Thr_{trans} \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

Subsequently, each truck verifies if the fingerprint contains enough entropy by computing the ratio of high bits. If the ratio is below certain threshold, the fingerprint is discarded and new fingerprint is generated in the next iteration.

5.4.4 Entropy Verification

To prevent an attacker from guessing the fingerprint, *Convoy* requires that the fingerprint exceed a certain amount of randomness. We define the *fingerprint weight* $w(F)$ as the fraction of *high bits* in a fingerprint F, capturing the amount of variation in the signal. Hence a fingerprint F with $w(F) = 0.5$ indicates a context that is most unpredictable to guess, as it has equal number of high and low bits. To capture this idea, we define a *fingerprint weight deviation* as $d_w(F)$. The following equations describe how $w(F)$ and $d_w(F)$ are computed.

$$w(F) = \frac{1}{|F|} \sum_i F[i], \qquad d_w(F) = 1 - 2\left|\frac{1}{2} - w(F)\right| \tag{5.3}$$

Hence a low weight deviation indicates that there are fewer contextual changes, making it easier for the attacker to guess the fingerprint. On the other hand, a high weight deviation indicates that there are more contextual changes, making it difficult for the attacker to guess.

Note that the maximum $d_w(F)$ is 1 when half of the bits are *high*.

Consequently, *Convoy* requires the committing vehicle (B in the example) to compute the fingerprint weight deviation and only transmit the commitment if $d_w(F) > Thr_w$ for a given threshold Thr_w.

5.5 Evaluation

We evaluate *Convoy* through experimentation with vehicles in real traffic scenarios. We first describe the experiment setup and then evaluate the effects of road conditions, leaving evaluation of traffic conditions for future work.

5.5.1 Experiment Setup

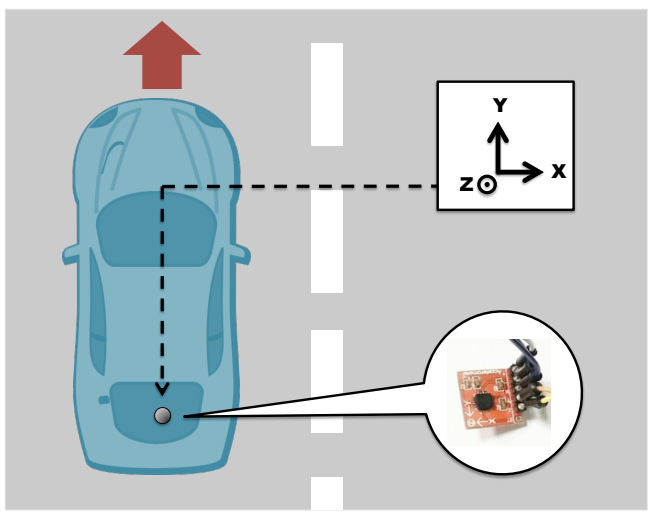


Figure 5.4: Illustration of experiment apparatus for evaluation. Y-Axis is parallel to the lane and Z-Axis is perpendicular to the road surface.

We experiment by driving two distinct vehicles (2014 Volkswagen Jetta and a 2012 Subaru Impreza) with trial driving segment spanning over six miles of highway by cruising at 65 mph. We only test the road condition by keeping the traffic condition consistent and delay the traffic condition analysis for future work. Each car was driven in two lanes, with two trials each, yielding a total of 48 miles worth of sensor data. We deployed a triple-axis MEMS accelerometer [49] (with a range of -3 to 3 g sampling at 5KHz) on an Arduino Uno board [10]

in the trunk of each car. The z-axis is normal to the road surface to measure road conditions, while the y-axis of the accelerometer points in the direction of travel to measure acceleration due to traffic conditions.

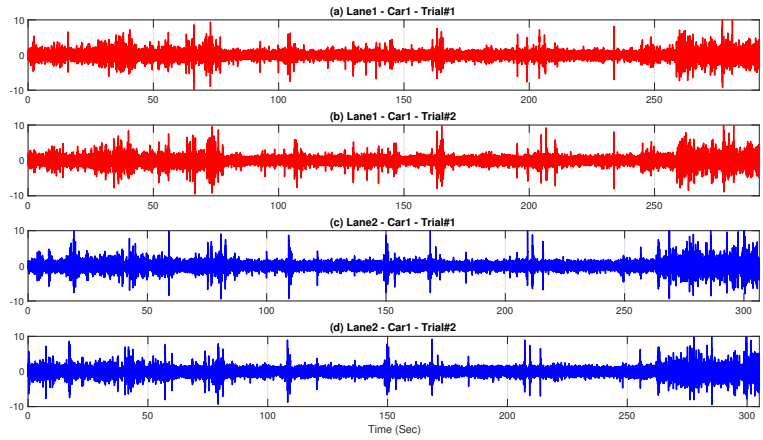


Figure 5.5: Subsection of accelerometer (Z-Axis) time series data ($\approx$ 5 minutes of drive at 65 mph) of adjacent lanes with two independent trials.

5.5.2 Fingerprint Similarity

We compare extracted fingerprints from the z-axis accelerometer to evaluate the feasibility of distinguishing between vehicles driving in different lanes, where each fingerprint has a length of 128 bits. Figures 5.5(a)–(b) and (c)–(d) exemplify the fingerprint similarities between vehicles traveling on the same lane (measured by two trials of the same car). However, comparison across the two pairs depict significant deviance, sufficient to distinguish two adjacent lanes. We discuss our results in three separate cases: similarity between different trials of the same vehicle in the same lane, between different vehicles in the same lane, and between the same vehicle in different lanes. This last case highlights the best-case scenario for an attacker, since the hardware is eliminated as a variable.

Similarity across trials of same vehicle in same lane. We show that the fingerprint pairs created from the same vehicle traveling on same lanes are consistent. We extracted fingerprints from accelerometer data which reflects bumpiness due to imperfection of the road. We repeated this on total of two vehicle models and report the fingerprint similarity of the aggregate result in Figure 5.6. As the figure shows, high fingerprint similarity is observed in different driving instances of same road with an average of 92.8%. We also note that this result would improve further with usage of lane control modules (such as the Adaptive Cruise

Control (ACC)) in a real scenario.

Similarity across vehicles in the same lane. As the same vehicle traveling in the same lane creates consistent fingerprints, we perform additional evaluation to confirm whether the fingerprint similarity is retained as we change vehicles. Again, we use all possible pairs of fingerprints created from accelerometer data. We report the resulting average fingerprint similarity result of 90.6% in the same figure. While the data trends show slight degradation, the fingerprints remain fairly consistent.

Similarity across lanes. We next compare fingerprint similarities for the same vehicle in adjacent lanes. Using the same vehicle minimizes the effect of mechanical variation and reflects a benefit for the attacker. We perform the fingerprint extraction and compare the fingerprint similarity between two different lanes traveled by the same vehicle and report the aggregate result of 81.6% in the same figure.

We also present a set of p-values that compares how fingerprint similarity compares between the Same-Car-Same-Lane (SCSL), Different-Car-Same-Lane (DCSL), and Same-Car-Different-Lane (SCDL) conditions, as depicted in Table 5.1. The comparison between Same-Lane conditions (SCSL vs. SCDL) yielded p-value of 0.60, showing that these two are not significantly different. However, the comparison between any Same-Lane conditions with Same-Car-Different-Lane condition (SCSL vs. SCDL and DCSL vs. SCDL) yielded 0.0008 and 0.003 respectively, showing significant difference in both comparisons.

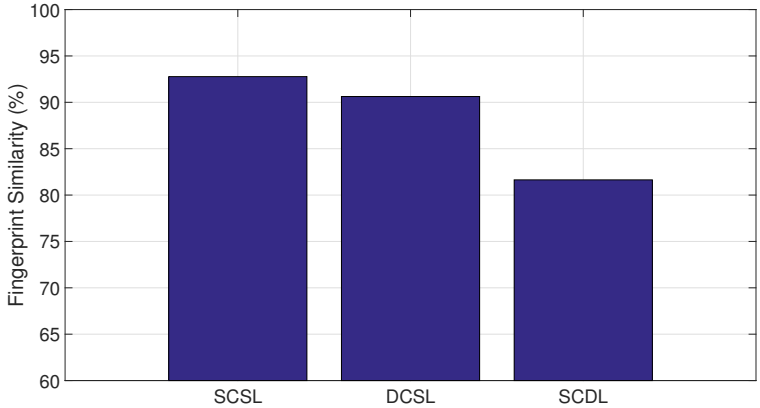


Figure 5.6: Comparison of fingerprint similarity due to road conditions for *Same-Car-Same-Lane (SCSL)*, *Different-Car-Same-Lane (DCSL)*, and *Same-Car-Different-Lane (SCDL)*

Comparison Pair		p-value
Same Car – Same Lane	Different Car – Same Lane	p=0.60
Same Car – Same Lane	Same Car – Different Lane	p=0.0008
Different Car – Same Lane	Same Car – Different Lane	p=0.003

Table 5.1: Paired t-test for comparison pairs from Figure 5.6.

5.6 Related Work

This section presents related work including platooning insider attacks, V2V wireless security, contextual authentication, and pothole detection.

Platooning insider attacks. There has been prior work in platooning security, especially on insider attacks. This category of research deals with abnormal behaviors from vehicles inside a platoon. An inside attacker can launch a collision induction attack by signaling its following vehicle to speed up while itself de-accelerating. This will abruptly reduce the distance between the attacker and its follower. As it is, the distance between adjacent vehicles in a platoon is usually small. Therefore, at high speeds, it is likely to cause an accident and also probably induce all the following vehicles to collide, leading to a massive pile up accident. Another issue is that rogue vehicles inside the platoon can reduce the efficiency of the platoon by increasing the distances between adjacent vehicles thereby, increasing the drag (wind pressure) and reducing the fuel efficiency. These insider attacks can be mitigated by techniques mentioned in the following papers [46]: each vehicle in a platoon models the expected behavior of the car directly preceding it, and detect any inconsistency between actual behavior and the expected behavior. Two or more attackers within a single platoon can also team up to implement a collusion attack. In this attack, some of the attackers within the platoon accelerate while some others strategically located break at the same times. This produces heightened oscillations in the platoon, thereby increasing its instability as demonstrated by Dadras et al. [44]. *Convoy* focuses on outsider attacks when malicious vehicles attempt to join a platoon. Our work serves as a complement to the existing countermeasures and contributes to the comprehensive protection of platooning.

WAVE/DSRC security. (location spoofing, Sybil attacks, voting, etc.) Many researchers have proposed using traditional DSRC (VAuth references 9,23,25)/ WAVE security mechanisms in previously published VANET papers. They leverage the PKI for authentication in vehicle to vehicle (V2V) communication. This scheme leaves the system vulnerable to spoofing and forging attacks such as the Sybil attack. To mitigate this threat, researchers

have proposed using a global reputation system [46], which acts as a mass surveillance check whereby if a vehicle is flagged on multiple occasions by several different vehicles, it can run a diagnostic check to test for system failures [141]. It can also get its license revoked by authorities.

Recurrent contextual authentication. Ambient contextual information has been leveraged for the purpose of authentication. Miettinen et al. studied secure pairing of IoT devices [99]. The proposed pairing scheme relies on the fact that co-present devices can sense similar ambient context, from which they can extract fingerprints as their shared secret. Another work [117] investigates secure communication through ambient audio. It proposes methods to effectively extract fingerprint from ambient audio and to establish a secure channel by using fuzzy-cryptography. *Convoy* tackles the authentication challenges in another emerging application scenario. We study how to obtain useful information from both the traffic and road conditions.

Pothole detection. Previous work [55] examines the feasibility of using solid-state sensors to detect road conditions. Specifically, they leverage measurements from accelerometers deployed on normal cars. The results show that it is effective to assess road surface conditions through the vibrations detected by those sensors. *Convoy* also uses road condition information. Instead of merely detecting potholes or bumps on road, we compare the road conditions experienced by different vehicles to generate entropy for authentication.

5.7 Discussion

We now present two main discussion points of *Convoy*.

Road conditions in different cities. The experiments were performed in relatively newer roads in California, which do not have considerable wear and tear. However, given that *Convoy* shows promising performance even with such conditions, we expect to find higher variations from road segments of cities subject to more severe weather conditions.

Sensor placement in trucks. While we report experimental results by driving two sedans, we note that the accelerometer readings from trucks will most likely yield similar results with trivial adjustments such as minor changes to the signal processing algorithm as well as a more careful sensor placement. We note that platooning trucks could place their sensors in locations more sensitive to road conditions and truck movements (e.g., perhaps below the chassis).

Pre-shared keys. One may propose trucks from same vendors to share keys in advance.

However, such solutions are not sufficient because of two reasons. First, truck platooning envisions supporting trucks on the road to form a platoon in an ad-hoc fashion regardless of their vendors. Second, even in the extreme case of platoon formation among trucks from same vendors, key pre-sharing approach is inherently vulnerable to insider attack, where a supposedly valid truck turns malicious and launches a ghost attack. *Convoy* addresses such problems because it provides the trucks supplemental guarantee of their physical arrangements.

5.8 Chapter Summary

We propose *Convoy* to secure trucks admissions into a platoon by verifying physical context. *Convoy* is novel because it leverages inherent randomness from road and traffic conditions to autonomously bootstrap a shared cryptographic key that is used by vehicles to securely bind physical context, or locality information to digital identifiers, or certificates. We implement and evaluate the *Convoy* fingerprint verification scheme against real-world driving data collected from two different vehicles, and demonstrate the feasibility of sufficiently differentiating between adjacent lanes using only single axis of an accelerometer data. As our future work, we plan to conduct more rigorous experiments covering longer segments with varying traffic conditions. We also plan to provide a robust defense against potential replay attacks on a targeted vehicles on specific road segments.

Chapter 6

Risks of Sensing

While sensor measurements may complement verification of physical relationships as shown in the above chapters, we also demonstrate that sensing may lead to potential security vulnerabilities of IoT devices and cyber-physical systems. In this chapter, we present how attackers may capture contextual information through device coordination and sensor fusion. Specifically, we present the feasibility of launching a side-channel (eavesdropping) attack by reconstructing intelligible speech signals, by fusing multiple non-acoustic IMU-based sensor data available in a smart home environment [66].

6.1 Problem Definition

Emerging technologies in the Internet of Things (IoT) give rise to wide deployment of pervasive networked sensors. This trend is evidently increasing as recently demonstrated [2, 103], projecting an IoT and global sensor market of $1.7 Trillion and $190.6 Billion, respectively, by 2021. As the number of IoT devices increase, sensors will surround us to monitor various parts of our lives at our homes, offices, and numerous other places.

While sensors contribute to numerous constructive applications, some of the recent research demonstrate the feasibility of launching side-channel attacks to leak privacy sensitive information. The following attacks infer sensitive information of a victim using only accelerometer data from a smart phone. *ACComplice* demonstrates how an attacker could infer a victim's location [69]. *ACCessory* infers a victim's keystroke presses on a smartphone [105]. *spiPhone* infers a victim's key presses on a computer keyboard when the smartphone is located nearby [94].

All of the aforementioned side-channel attacks focus on extracting private information

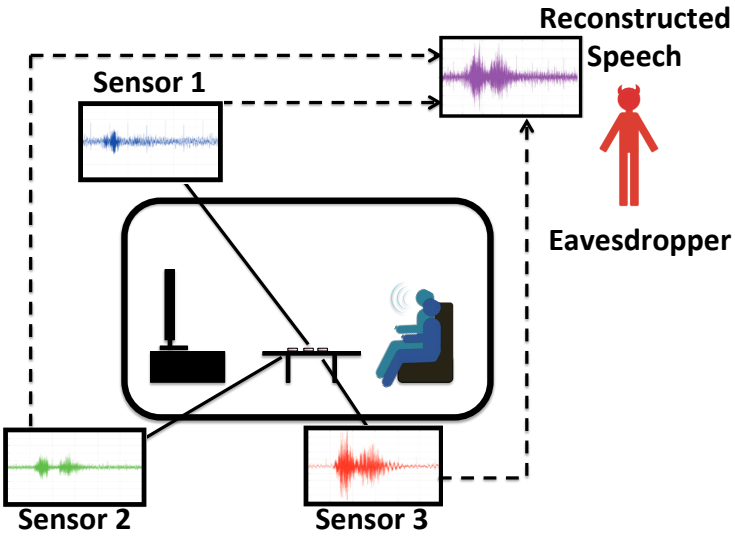
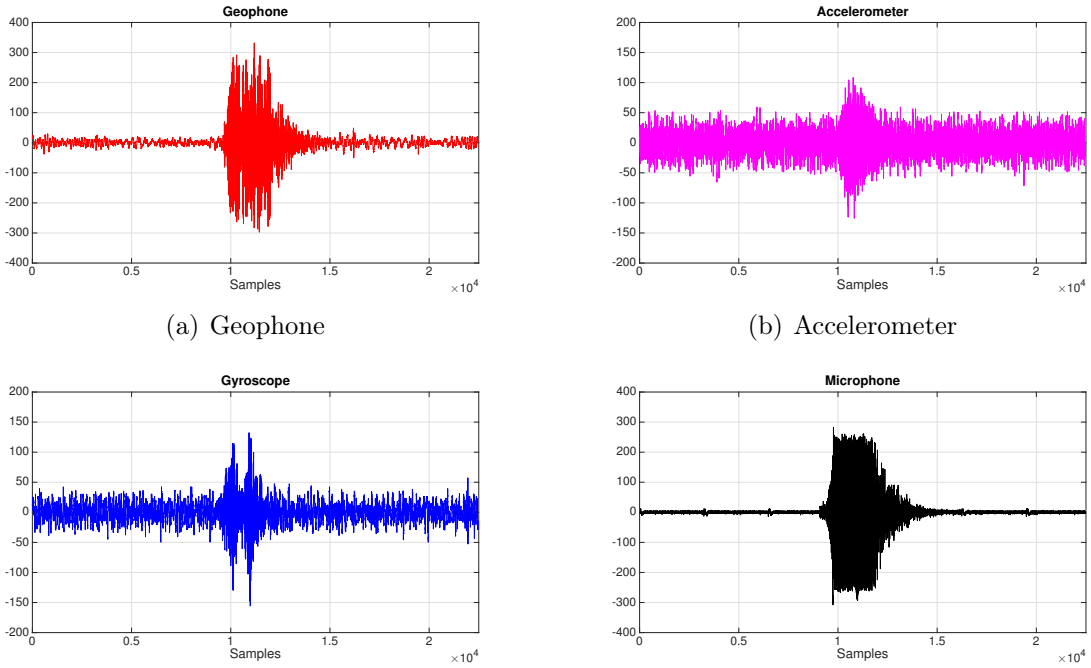


Figure 6.1: Example scenario where non-acoustic sensors embedded in IoT devices are "listening" to conversations.

from an individual sensor. However, the expected penetration of IoT devices into our homes and workplaces inspires us to consider additional threats due to wide deployment of sensors, including environmental and activity sensors used for common IoT applications. Beyond the already prevalent sensing capabilities of smartphones, smart watches, and tablets, we are now seeing activity sensors (such as accelerometers and gyroscopes) deployed in smart TV remotes and gaming controllers (e.g., Wii remote, PS4 and Xbox controllers) [54, 129]. In addition, we are seeing environmental and structural sensors (such as temperature, humidity, and geophone sensors) in smart buildings and smart cities [81, 106, 107, 130]. Many of these sensors are widely deployed in experimental and generic wireless sensor boards for multi-purpose sensing [86, 92, 137], and we expect their deployment and inclusion in commercial services to increase dramatically based on the market projections mentioned above.

With such wide deployment of sensors in IoT devices, we find that a large portion of research community has concentrated on finding and defending against vulnerabilities of individual sensors or devices, and what the posed risks are for the users. However, we are more interested in exploring new vulnerabilities if an attacker compromises data collected from multiple devices. Specifically, we pose the question – what unforeseen information can one extract from fusion of these sensors across networked devices?

In search for the answer to the above question, we present ***PitchIn*** to demonstrate the feasibility of achieving the seemingly unrealizable goal of reconstructing an *intelligible speech*



(a) Geophone (b) Accelerometer

(c) Gyroscope (d) Microphone

Figure 6.2: Time series plots of geophone, accelerometer, gyroscope, and microphone of the word "one" sampled at 8 KHz.

signal by fusing non-acoustic sensor data collected from a network of nodes. Specifically, we consider scenarios of potential security breaches of a smart home/office's gateway or in service provider's database, which has logs of sensor data from victim's IoT devices. Such breaches have been witnessed in many real-world examples recently [20, 25, 63]. Hence, the attacker does not have to compromise individual devices equipped with sensors in victim's home or office to gain access to the sensor data. We illustrate an example scenario depicted in Figure 6.1.

Traditionally, non-acoustic sensors such as geophones, accelerometers, and gyroscopes are thought to be unresponsive to acoustic signals, as they are designed to capture motion signals (vibrations, movements, and tilt angles, respectively). However, we find from our experiments, along with the findings from related work, that when exposed to sound waves, the sensors vibrate to output minuscule signals, sufficient to be processed to reconstruct intelligible acoustic signals [97, 142]. Figure 6.2 depicts the time series plots of non-acoustic sensors such as a geophone, an accelerometer (x-axis), and a gyroscope (x-axis), when sampled at 8 KHz[1].

[1]We note that we only use the x-axis of accelerometer and gyroscope throughout this thesis for simplicity, but the axis can be interchanged or combined with other axes.

We also show microphone data for comparison.

Unfortunately, a sampling frequency of 8 KHz is much higher than the typical rate at which these motion sensors are configured to be sampled at in commercial devices. Obtaining intelligible speech signals, however, require a high sampling frequency, with a minimum of 5 KHz [110], while telephones and CDs are sampled at 8 KHz and 44.1 KHz, respectively [42, 93] for higher quality audio. Hence, an attacker cannot recover an intelligible speech from sensor data of a single device.

To increase the overall system sampling frequency, *PitchIn* builds upon the idea of Time Interleaved Analog-Digital-Conversion (TI-ADCs) [6, 7], which is a method to parallelize the sampling task with multiple ADCs with temporal offset. *PitchIn* extends this idea to create **Distributed TI-ADCs** so that the reconstructed signal, which we refer to as the *Amalgam signal*, has an overall effect of being sampled at a high sampling frequency. In reality, however, each node is sampled at a much lower sampling frequency. Hence, each node is "*pitching in*" to contribute to the *Amalgam* signal.

Even with the high overall *Amalgam* signal sampling frequency thanks to *PitchIn*'s Distributed TI-ADC, achieving intelligibility from the reconstructed *Amalgam* signal is extremely challenging because fusion of sensor data creates mismatches in amplitude alignments and causes distortions. Hence, we transform the signals using different signal processing techniques (e.g., normalization and denoising) to reconstruct a final speech signal that can be interpreted by humans.

We evaluate the intelligibility of *PitchIn* via a user study (approved by our Institutional Review Board (IRB)) by reconstructing two sets of *Amalgam* signals constructed of varying number sensors sampled with per node sampling frequency of 500 Hz and 1 KHz.

6.2 Background

In this section, we present relevant background information on how motion sensors function. We also present the main idea of interleaved ADC, and how it increases the overall sampling frequency. We then present how speech signals can be reconstructed from motion sensors.

6.2.1 Sensors

Even though the device physics of each sensor modality varies, all sensors operate under the same principle: they capture physical signals and transform them into electrical signals.

Specifically, we address how a geophone, an accelerometer, and a gyroscope perform such transformations.

Geophone. Geophone captures mechanical vibrations that travel through solid media [38]. As illustrated in Figure 6.3, a geophone consists of proof mass, magnet, and coil. As mechanical waves reach the base of a geophone, small vibrations cause the base magnet to vibrate. Subsequently, an electrical coil attached to the proof mass experiences changes in magnetic flux. Such events translate the mechanical signal to voltage induction which is output as an analog signal. As geophones are tuned to capture longitudinal mechanical waves, it is no surprise that they capture sound waves as well. Vibrations from sound waves induce small vibrations of the sensory mechanism, so acoustic waves are registered as small but detectable signals in the analog output.

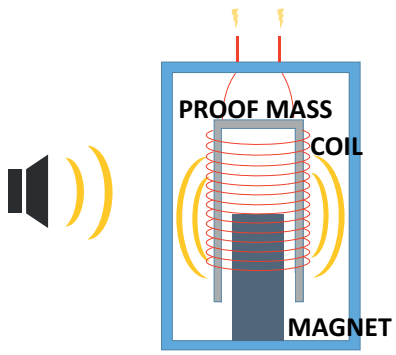


Figure 6.3: Illustration of how a geophone translates physical movements into voltage.

Accelerometer. Similarly, accelerometers capture mechanical vibrations through its sensing axes [48, 49]. As depicted in Figure 6.4, a commercial Micro Electro Mechanical Systems (MEMS) accelerometer has physical structures that allow movement in a proof mass to be translated to voltage signals by change in capacitance. As the MEMS sensor accelerates along the axis of interest, a fictitious inertial force shifts the proof mass to swing between springs. The change in the distance between the metal plates results in the change in capacitance, yielding the analog signal change which can be mapped to the acceleration value using a predetermined conversion factor.Most MEMS accelerometers include three such mechanisms to realize X, Y, and Z orientations.Since acoustic pressure waves exert a force on the proof mass, small vibrations occur and yield an analog signal output that would otherwise be interpreted as acceleration.

Gyroscope. MEMS gyroscopes also have a similar structure to that of MEMS accelerometers [126]. As depicted in Figure 6.5, as a gyroscope is rotated, the proof mass rotates

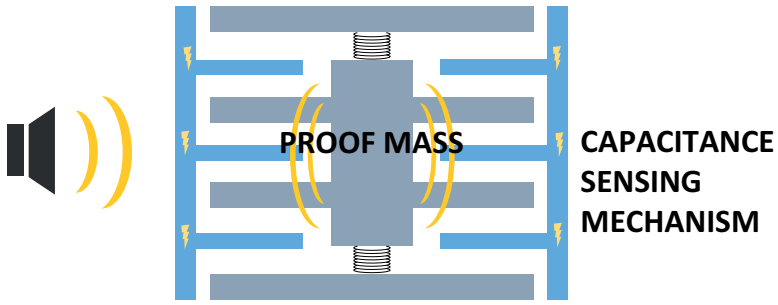


Figure 6.4: Illustration of how an accelerometer translates physical movements into voltage.

as a result of the fictitious Coriolis force. This force is analogous to that of inertial force in translation. As metal plates rotate as a response, the capacitance change is registered as an analog signal. As acoustic waves come in contact with a MEMS gyroscope, small vibrations that reach the proof mass also create vibration along the rotating axis, translating to electrical signals through capacitance.

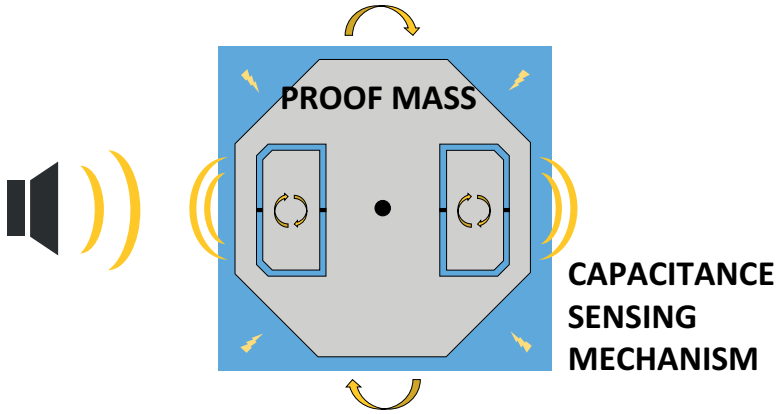


Figure 6.5: Illustration of how a gyroscope translates physical movements into voltage.

IoT Devices for Different Applications	Gaming Controller (Wii Remote)	Smart TV Remote	Smartphones/ Smartwatches/ Tablets	Earthquake Detection Device	Footstep Monitoring Device	Structural Health Monitoring Device	IoT Sensor Board
Sensors	Accelerometer/ Gyroscope	Accelerometer/ Gyroscope	Accelerometer/ Gyroscope	Geophone	Geophone	Geophone/ Accelerometer/ Gyroscope	Geophone/ Accelerometer/ Gyroscope

Table 6.1: IoT devices used for different applications and the corresponding sensors embedded in the devices.

6.2.2 Sensors Embedded in IoT Devices

Different IoT devices have various sensors depending on their applications. We highlight example IoT devices that include geophone, accelerometer, or gyroscope. Different devices sample these sensors at varying frequencies depending on the application. Higher sampling frequency captures more information resulting in more accurate representation of the signal, but at a cost of higher computational and energy costs. Table 6.1 depicts some of the IoT applications and the corresponding sensor modalities.

Controllers for gaming consoles (e.g., Wii Remote, PS4 Dualshock4 Controller, Xbox Controller) embed accelerometers and gyroscopes to detect user motion for dynamic gaming experiences [129]. Similarly, smart TV remotes embed sensors for user gesture recognition and identification [54]. These devices sample sensors on the order of 100 Hz.

Mobile devices such as smartphones, smart watches, and tablets embed a large number of sensors, including accelerometers and gyroscopes, used for various applications (e.g., activity/gesture recognition, gaming, etc). Mobile OSes such as iOS and Android restrict the sampling frequencies of these sensors to a maximum of 200 Hz.

IoT devices used for monitoring applications also embed geophones, accelerometers, and gyroscopes. Earthquake detection devices leverage geophones to measure and analyze vibrations. Indoor footstep monitoring systems of occupants also make use of geophones [106, 107]. Structural health monitoring devices make use of the above three sensors to monitor the condition of buildings and/or bridges [81, 130]. These devices sample on the order of 1 KHz.

Furthermore, the industry and academia are pushing forward to deploying sensor platforms that integrate a suite of general-purpose sensors driven by various environmental sensing applications [86, 92, 137]. These devices enable varying sampling frequencies depending on the application (on the order of tens of KHz).

6.2.3 Time Interleaved ADC

Time Interleaved Analog-Digital Conversion (TI-ADC) has been explored to acquire high sampled data on resource-constrained systems. The main idea behind TI-ADC is that while each ADC is bounded by a relatively low sampling frequency, it is possible to increase the effective sampling rate by using multiple ADCs in parallel. Specifically, a set of multiple ADCs are placed at different temporal points to sample at a low frequency [6, 7]. Subsequently, software recombines the pieces of sampled data. Assuming time synchronization, TI-ADC allows effective sampling frequency to increase by a factor of the number of ADCs. This is

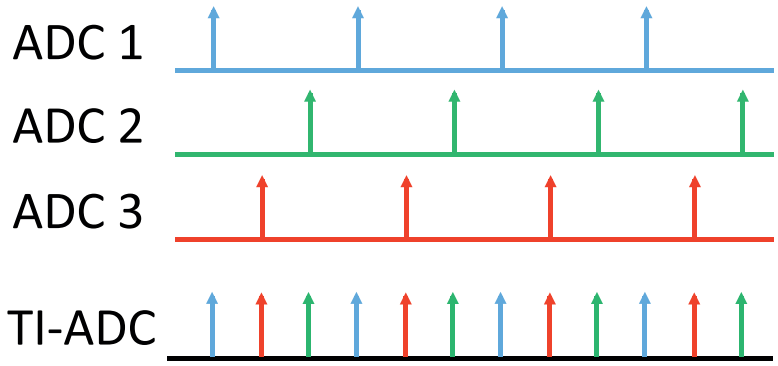


Figure 6.6: Illustration of how TI-ADC increases the overall sampling frequency by leveraging multiple ADCs in parallel with temporal offset.

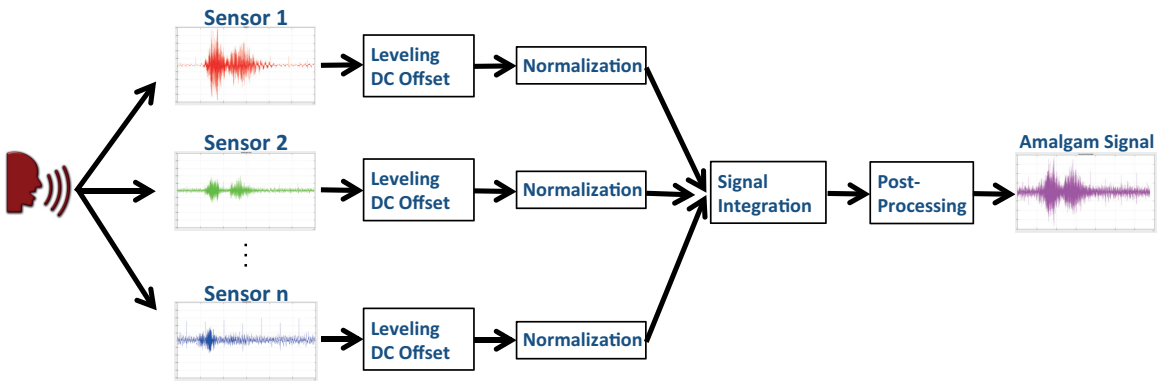


Figure 6.7: System overview diagram of *PitchIn* speech signal reconstruction.

depicted in Figure 6.6. *PitchIn* builds upon this idea, but rather than using multiple ADCs on a single physical system, we treat distributed devices in a network as "virtual" ADCs.

6.2.4 Speech Intelligibility

Two of the main factors contribute to achieving speech intelligibility – (1) sampling frequency and (2) contextual information. Human auditory systems process acoustic signals up to 20 KHz. Due to the Nyquist sampling theorem – which defines minimum required sampling rate for the signal [119] – audio files on CDs are created using a sampling rate of 44.1 KHz to avoid distortion [42, 93]. We also note that minimum sampling frequency of 5 KHz is required for intelligibility of human speech signals [110].

Another factor to consider is the context within speech. Speech recognition by humans is known to be a complex experience that subconsciously perceives words that make best sense within the given context. When a distorted signal is presented to human perceptual system, it

is known to perform much better when the context of the information is also presented [134]. Inspired by the human speech recognition, automatic speech recognition (ASR) tools also use language models to increase the recognition accuracy [80].

In this thesis, we take into consideration how sampling frequency from each sensor affects reconstruction of speech signals. Furthermore, we also take into consideration of contextual information when designing our user study to reflect the reality of speech recognition performed by humans.

6.3 Adversary Model

We now present the threat model of *PitchIn*. Specifically, we present the goals and capabilities of the attacker as well as the assumptions made. The main goal of the attacker is to launch a successful eavesdropping attack on victim's spoken verbal communications in his/her home, office, conference rooms, etc. Specifically, we consider an *offline attack* made possible by potential breaches of recorded sensor data from a gateway in a smart home or service provider's database, often encountered in many real-world incidents [20, 25, 63].

Furthermore, we note that in practice, each IoT devices samples their non-acoustic sensors at a low sampling rate (<1 KHz, as presented in Section 6.2.2). Consequently, resulting signals from individual sensor produce non-intelligible sound. We assume an attacker without the capability of remotely modifying the sampling frequencies of the sensors in each of the devices. Hence, the attacker may interleave multiple signals (with low sampling rate per signal) captured by different devices to achieve a *Amalgam* signal that has an overall effect of a single device with a high sampling rate, increasing the intelligibility.

We also assume that the attacker does not have access to a microphone data, which is usually sampled at a high sampling rate (>5KHz [110]). Otherwise, the attacker will directly make use of the microphone data instead of the non-acoustic sensor data, eliminating the need to interleave signals of different devices.

6.4 Design and Implementation

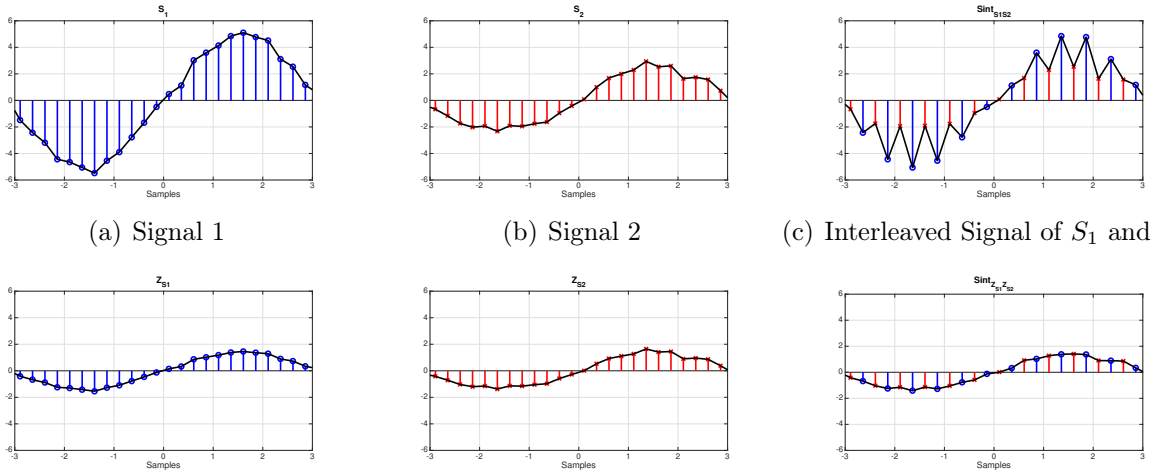
We now discuss the implementation details of reconstructing an intelligible *Amalgam* signal by fusing data collected from a network of sensors. We first present an overview of the *Amalgam* signal generation, and then discuss the details.

6.4.1 Design Overview

To construct *Amalgam* signals from different sensors, *PitchIn* leverages a distributed form of Time Interleaved Analog-Digital Conversion (Distributed TI-ADC). This is to generate an effect of high sampling frequency (Fs_{Amal}) signal from a fusion of multiple sensor data that are sampled at low per-node sampling frequency (Fs_{sensor}). However, distributed TI-ADC requires addressing difficult challenges to produce an intelligible speech signal. Figure 6.7 depicts the flow chart diagram of *PitchIn Amalgam* generation steps. First, each sensor data is sampled locally with its low Fs_{sensor}. Then each individual signal is leveled to account for DC offset mismatches that occurred during the ADC phase. Subsequently, individual signals are normalized to be aligned because different physical sensors lead to gain mismatches. We then leverage distributed TI-ADC to interleave different signals into one *Amalgam* signal and then perform post-processing such as interpolation and denoising.

6.4.2 Implementation

We discuss in detail how *PitchIn* addresses the following main challenges: *leveling DC offset*, *gain normalization*, accounting for *temporal offset mismatches*, and *post-processing*.



(a) Signal 1 (b) Signal 2 (c) Interleaved Signal of S_1 and S_2

(d) Signal 1 Normalized (e) Signal 2 Normalized (f) Interleaved Signal of Z_{S1} and Z_{S2}

Figure 6.8: A toy example of amplitude normalization and its effects.

6.4.2.1 Leveling DC Offset

Data sets from different sensors may have distinct DC offset, or average value offset from 0 volts [27], because of the way that analog signals are converted to digital signals. With the aggregated data from all the nodes, *PitchIn* reconstructs the *Amalgam* signal by first leveling the DC offset. Leveling the DC offset is important to speech intelligibility because the DC offset contributes to distortion or reduced audio volume.

6.4.2.2 Gain Normalization

Data sets from different sensors also exhibit different amplitude levels due to the differences in how each sensor captures the vibrations from the sound signal and the differences in the amplification level before going through the ADC. Amplitude normalization is imperative for *PitchIn* to reconstruct intelligible speech signal by fusing different sensor readings. Figure 6.8 depicts a toy example that illustrates this concept. Figure 6.8(a) and 6.8(b) depict two signals, S_1 and S_2, respectively, exemplifying noisy sensor readings of a sinusoidal signal with non-aligned amplitudes. Figure 6.8(c) depicts the resulting interleaved signal, $Sint_{S1S2}$, when no amplitude normalization is performed. We note that the resulting signal is heavily distorted.

However, we show the effect of normalization with the remaining subfigures. Figures 6.8(d) and 6.8(e) depict Z_{S_1} and Z_{S_2}, which are output of Z-Score normalization of S_1 and S_2, respectively. Figure 6.8(f) depicts the resulting interleaved signal, $Sint_{Z_{S_1}Z_{S_2}}$ of the normalized signals, Z_{S_1} and Z_{S_2}. As depicted from this figure, the resulting signal has a high resemblance to the original sinusoidal signal.

While other types of normalization methods may be applied, we leverage Z-Score because it computes the statistical quantification of how much each score is distant from the mean in terms of standard deviations. Within a sensory modality, the signal to noise ratio of audio signal is expected to be similar between the sensors. This allows usage of Z-scores to project the signals in a statistically normalized space, where the amplitude of the signals in all the sensors will be aligned to one another based on signal to noise ratio. The normalized value of Z-Score Z_{S_i} is computed for data S_i from the ith sensor that has a known mean μ_i and standard deviation σ_i is computed as $Z_{S_i} = (S_i - \mu_i)/\sigma_i$.

6.4.2.3 Accounting for Temporal Offset Mismatches

Different devices start sampling their sensors at different times. We note that to achieve the best results for the distributed TI-ADC, each device has to sample at a regular interval relative to each other, resulting in a perfectly interleaved signals. This increases the *Amalgam* signal sampling frequency by n times, where n is the number of sensors. As a proof-of-concept, we demonstrate this with experiments in Section 6.5. However, achieving a perfectly interleaved data is extremely infrequent in practice. Rather, the temporal offset is close to being modeled as random. We demonstrate, however, that even with such limitations, *PitchIn* obtains reasonable recognition accuracy depicted in Section 6.5.3.2.

6.4.2.4 Post-processing

Interpolation. Once available data points are collected, spline interpolation is used to estimate the original signal. We interpolate the signal to output a *Amalgam* signal with a sampling frequency of 40 KHz. This method uses pieces of polynomials to estimate the region with no signal. Because spline interpolation has no restriction on how available data points are spaced, it is appropriate to use especially in the current implementation where data points may be available at random temporal offsets.

Filtering. We then perform high-pass filtering to the normalized signal to remove the transient noise. We leverage a fourth-order Butterworth filter [37] with a cutoff frequency at 300 Hz. Butterworth filter design uniformly preserves the passband frequency, while attenuating stopband frequencies.

6.5 Evaluation

We now describe the implementation and evaluation details of *PitchIn* eavesdropping attack. We first present the experiment setup and the implementation details. We then present and analyze different evaluation scenarios.

6.5.1 Experiment Setup

Apparatus. We implement *PitchIn* by interfacing the sensors with Arduino Uno boards [10]. Each Arduino board interfaces with one distinct sensor, namely a geophone, accelerometer, or gyroscope. For ground truth, we also interface an Arduino with a microphone. The apparatus

is depicted in Figure 6.9. The SM-24 geophone [38] is designed to detect ground movement and translates to an output voltage. The ADXL-335 three-axis MEMS accelerometer [49] measures and creates signals to represent the acceleration experienced by the sensor in the range of -3 to 3 g. The LPY403AL two-axis gyroscope [126] measures and outputs signals for the angular velocity of the pitch (X) and yaw (Z) axes in the range of -30 to 30 degrees per second. Each sensor is amplified in hardware using two operational amplifiers [73] and then fed into the Arduino's ADC. We refer to each of the board-sensor combinations as a *node*

The Arduino Uno board uses an 8-bit ATmega328P microprocessor [26]. It has 32 KB flash memory, 2 KB SRAM, and 1 KB EEPROM and a clock speed of 16 MHz. It has six analog interface pins. The single ADC has a resolution of 10 bits and output voltage range of 0 to 5 Volts. In our work, we modify the Arduino setting to range from 0 to 3.3 Volts to match the maximum output voltage of the sensors.

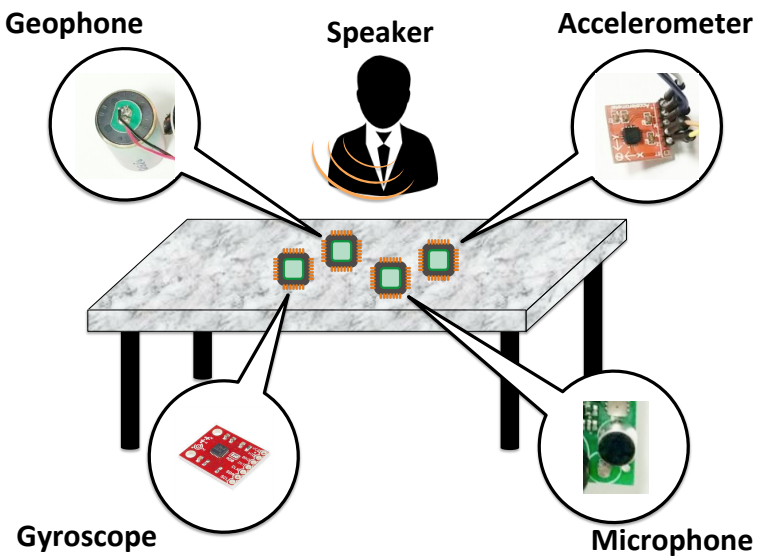


Figure 6.9: Experimental apparatus with a geophone, an accelerometer, a gyroscope, and a microphone.

Names of People	Joseph	Catherine	Thomas	Jefferson	Elizabeth	Michelle	Anthony	Emmanuel	Hilary	Patrick
Cities	Atlanta	Los Angeles	New York	San Francisco	Washington D.C.	Paris	London	Moscow	Tokyo	Hong Kong
Companies	Apple	Microsoft	Google	Facebook	Amazon	Comcast	Tesla Motors	Starbucks	Walmart	United Airlines
Numbers	One	Two	Three	Four	Five	Six	Seven	Eight	Nine	Ten

Table 6.2: WG_1, WG_2, WG_3, and WG_4 of *names of people*, *cities*, *companies*, and *numbers (1 to 10)*, respectively.

Data Collection. Each node logs data on a microSD card, leveraging Arduino Ethernet Shield [8]. We make use of SdFat Analog Bin Logger library [9] to enable low latency SD card writes so the Arduino can write while sampling at such a high frequency.

We place the apparatus on a desk about a meter away from the person speaking (henceforth called speaker). The speaker's average Sound Pressure Level (SPL) is 85 dB, a typical "presentation-level" volume. We measure the average SPL using SkyPaw's dBMeter app on an iPhone 6 [122] positioned close to the speaker. The speaker is a male and a fluent but non-native English speaker.

User Study Process. The goal of this study is to determine the intelligibility of the reconstructed *Amalgam* signals. Participants were given instructions to transcribe recordings of different words. The participants were given an additional information of the word group that each recording belongs to. There are four word groups of ten words, WG_1 constituting names of *people*, WG_2 constituting names of *cities*, WG_3 constituting names of *companies*, and WG_4 constituting *numbers* from one to ten. The additional information serve to provide contextual information synonymous to context within speech (e.g., words in a sentence), reflecting the reality of how humans perform speech recognition [134]. The words are listed in Table 6.2.

We recruit a total of 230 participants, and presented randomized words so that each participant does not listen to the same word from different signals. Hence, each data point in the figures of this section consists of 230 transcriptions. The participants were recruited via Amazon Mechanical Turk [24]. We performed the user study after receiving approval from our Institutional Review Board (IRB) and complied to the IRB's recommendation.

6.5.2 Non-Acoustic Sensors

Before presenting the *Amalgam* construction, we first evaluate how each of the individual non-acoustic sensors respond to human speech, and how the intelligibility varies corresponding to their sampling frequencies, F_s. We compare the results to microphone as a baseline. Figure 6.10 depicts the recognition accuracy of the non-acoustic geophone, accelerometer, and gyroscope sensors sampled at $F_s = 8$ KHz, compared to the baseline case of a microphone also sampled at the same frequency. This figure clearly depicts the fact that the non-acoustic sensors respond to speech signals, yielding non-negligible accuracies depicted in the figure. As expected, the microphone yields a highest accuracy (94%).

To provide a better understanding of these signals and deeper insight into our results, we

have posted audio and video clips at http://mews.sv.cmu.edu/research/pitchin/. The video clips show spectrogram reconstructions of the spoken words “apple" and “seven", using the open source audio editor Audacity. We strongly advise the readers to view the video clips together with the figures in this section.

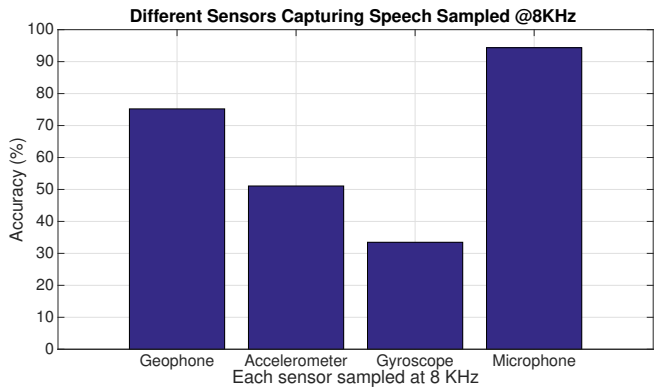


Figure 6.10: Non-acoustic sensors capture speech signal when sampled at a high rate (F_s=8KHz)

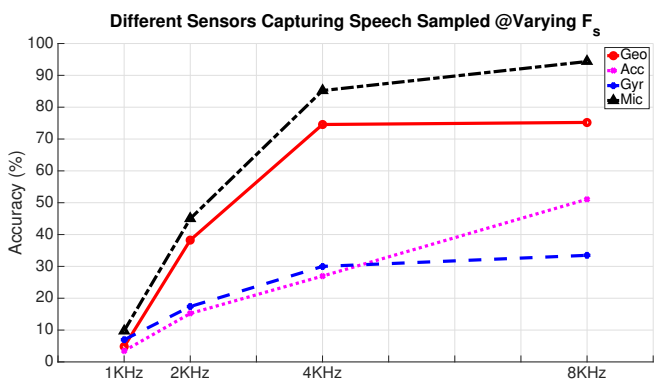
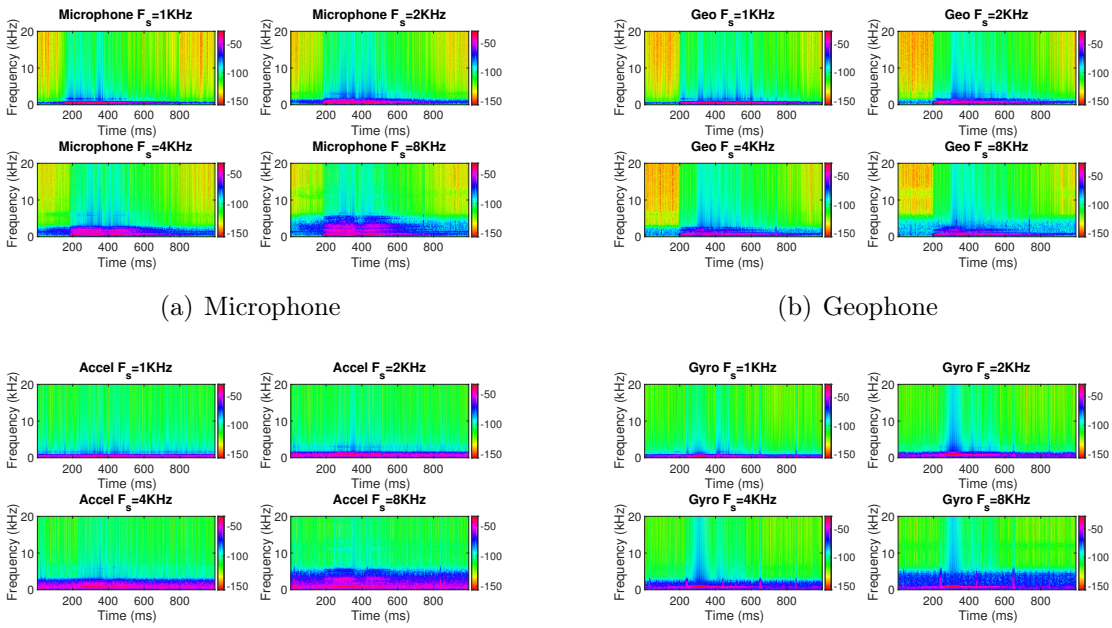


Figure 6.11: Recognition accuracy increases as F_s increases for each sensor.

We further investigate these sensors to test the relationship between the recognition accuracy (i.e., intelligibility) and the sampling frequency, F_s, depicted in Figure 6.11. As depicted from this figure, we note the trend of increasing recognition accuracy as F_s increases from 1, 2, 4, and 8 KHz. We also note that the accuracy is extremely low for all sensors when F_s=1 KHz, including the microphone (4.7%, 3.5%, 7.0%, and 9.8% for geophone, accelerometer, gyroscope, and microphone, respectively). Hence, we highlight that intelligibility decreases significantly as the sampling frequency decreases. Figure 6.12 depicts spectrograms of corresponding signals that yield the results evaluated in Figure 6.11. Furthermore, we



(a) Microphone

(b) Geophone

(c) Accelerometer

(d) Gyroscope

Figure 6.12: Spectrogram of single microphone, geophone, accelerometer, and gyroscope, each sampled at different F_s (Evaluated in Figure 6.11). We strongly advise the readers to view this figure in color.

demonstrate statistical significance of the results with paired t-test (along with t-test results of all following evaluations in this section) in Tables 6.3, 6.4, and 6.5 at the end of this section.

6.5.3 *Amalgam* Evaluation

We evaluate *Amalgam* signals constructed of fused sensor data. We first present the results of a proof-of-concept when sensor fusion is performed by interleaving signals with a regular temporal offset. We then present the results when we relax this assumption, more closely resembling the real-world scenarios. We also present an idea of fusing sensor data across sensor modalities.

6.5.3.1 Ideal Temporal Offset

We test the effects of achieving a higher *Amalgam* sampling frequency $F_{s_{Amal}}$ as we increase the number of nodes that "pitch in" to constructing the *Amalgam* signal. We report two sets of experiments as following. In the first experiment, we fix the per node sampling frequency,

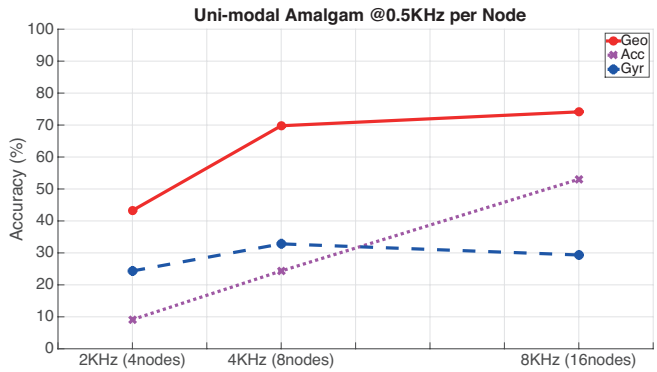


Figure 6.13: *Amalgam* signals constructed with F_s=500 Hz. Recognition accuracy of each *Amalgam* signal increases as $F_{s_{Amal}}$ increases from 2, 4, and 8 KHz by varying number of nodes from 4, 8, and 16.

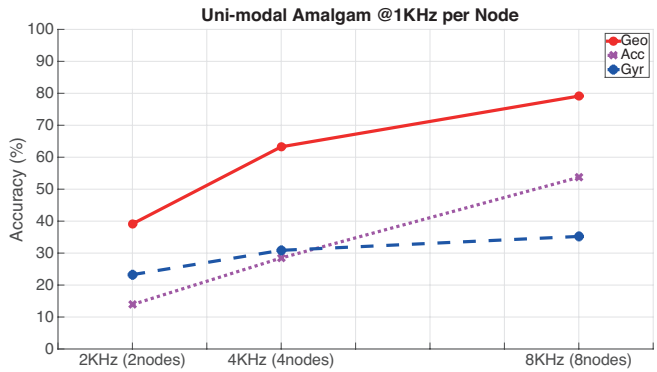


Figure 6.14: *Amalgam* signals constructed with F_s=1 KHz. Recognition accuracy of each *Amalgam* signal increases as $F_{s_{Amal}}$ increases from 2, 4, and 8 KHz by varying number of nodes from 2, 4, and 8.

F_s=500 Hz, and vary the number of nodes to 4, 8, and 16. Similarly, in the second experiment, we fix F_s=1 KHz and vary the number of nodes to 2, 4, and 8. Both experiments yield $F_{s_{Amal}}$ of 2 KHz, 4 KHz, and 8 KHz. Figures 6.13 and 6.14 depict the two experiments, respectively. We defer the discussion of how we "simulate" different sensor data from a single physical sensor readings for each of these sensors in Section 6.5.3.3. Figures 6.15 and 6.16 depict corresponding representative sprectrograms, respectively.

In both experiments, the trend of increasing recognition accuracy with increasing $F_{s_{Amal}}$ is preserved, similar to the non-*Amalgam* findings depicted in Figure 6.11. More specifically, the accuracy (i.e., intelligibility) significantly increases within most sensor modalities, yielding accuracies as high as 79%, 53%, and 35%, for geophone, accelerometer, and gyroscopes,

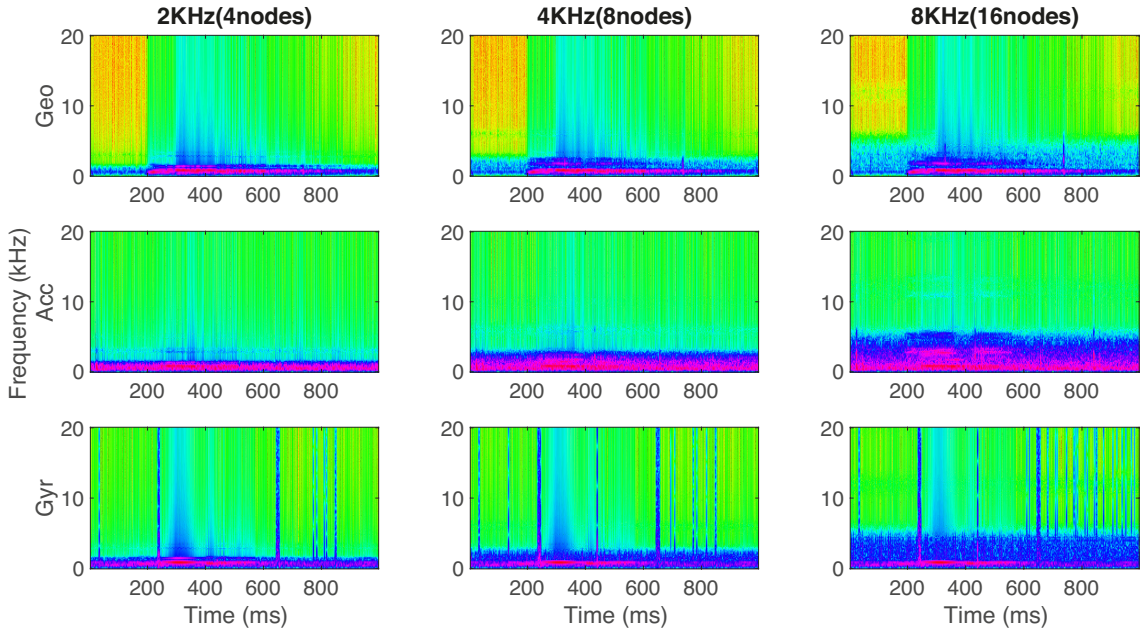


Figure 6.15: Spectrogram of *Amalgam* signals constructed with per node $F_s = 500$Hz. $F_{s_{Amal}}$ increases from 2, 4, and 8 KHz when varying number of nodes from 4, 8, and 16, respectively (Evaluated in Figure 6.13). We strongly advise the readers to view this figure in color.

respectively. We note that these numbers may significantly empower the attacker, as any additional information to the attacker is a gain when launching eavesdropping attacks, potentially posing serious threat to the victims. As an analogy, most people would feel uncomfortable or even threatened if 79% of their phone conversations are eavesdropped.

6.5.3.2 Practical Temporal Offset

Recall that the aforementioned results assume a regular temporal offset, which inherently results in the best case scenario for the *PitchIn* attack. However, in reality, temporal offset may be randomly distributed among devices. We investigate this aspect by exploring how varying temporal offset affects recognition accuracy.

To provide an intuition, we provide five different temporal offsets of four nodes sampling different gyroscopes. Figure 6.17 illustrates pictorial representation of a spectrum of varying temporal offsets (i.e., sampling patterns) from the worst case to the best case scenario. (a) depicts the situation when all four nodes are sampling exactly at the same time (hence the worst case scenario). (b) and (c) depict the situations when two of the nodes are sampling at the same time. Specifically, (b) depicts an example where there is not too much information gain from the temporal offset due to samples being clustered. We note that (c) resembles

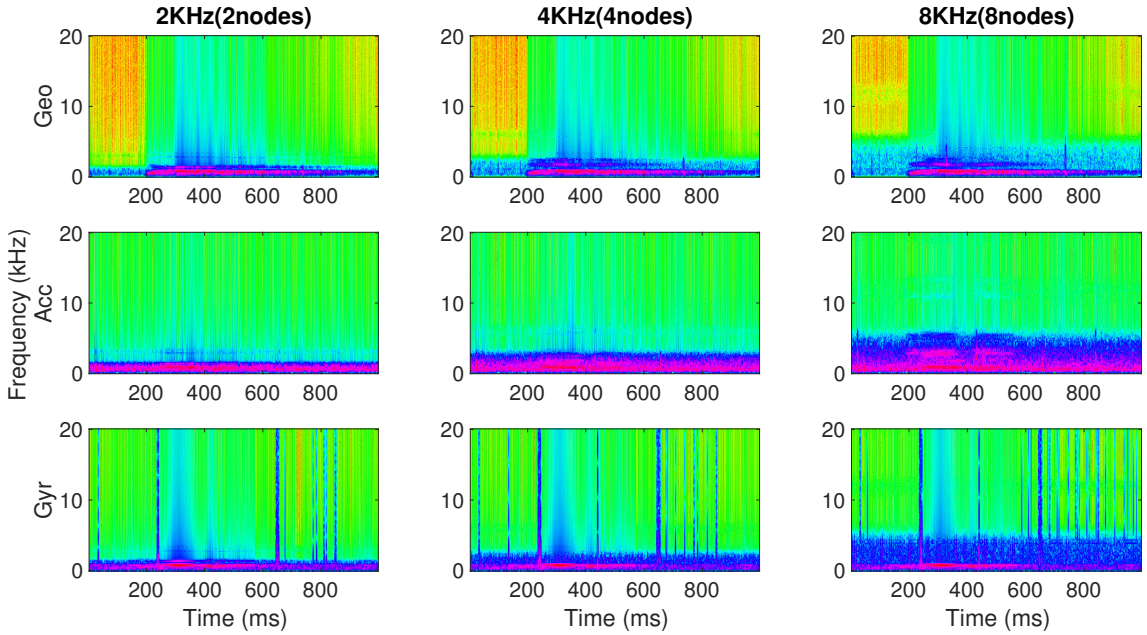


Figure 6.16: Spectrogram of *Amalgam* signals constructed with per node F_s = 1KHz. $F_{s_{Amal}}$ increases from 2, 4, and 8 KHz when varying number of nodes from 2, 4, and 8, respectively (Evaluated in Figure 6.14). We strongly advise the readers to view this figure in color.

the situation synonymous to when two nodes are sampling at an evenly distributed interval. (d) depicts the situation when four nodes are sampling at different times, but are not evenly distributed. Hence, the samples are more distributed, allowing larger temporal coverage. (e) depicts the situation when four nodes are sampling at an evenly distributed time (hence the best case scenario). We denote these as *Sample Scenarios (a)* through *(e)*.

Figure 6.18 depicts the recognition accuracy of (a) through (e) for four gyroscope sensors with F_s=1 KHz. We chose gyroscope to demonstrate the lower bound of recognition accuracy among the sensors (as seen from Figure 6.10). It is interesting to note that the recognition accuracy increases from (a) to (e), from 7% to 30%), which justifies the spectrum of varying temporal offsets from worst to best case scenario. Furthermore, we note that (c) yields roughly twice the accuracy of (a) and half of (e).

While scenarios (b), (c), and (d) are each single instances of temporal offset of these four sensors in between worst and best case scenarios (i.e., (a) and (e)), this example serves to demonstrate the trend of increasing recognition accuracy as temporal offset lies in between the two extremes.

We also present an example to provide an intuition of how “random" temporal offset still contributes to reasonable recognition accuracy by providing an example. The accuracy of

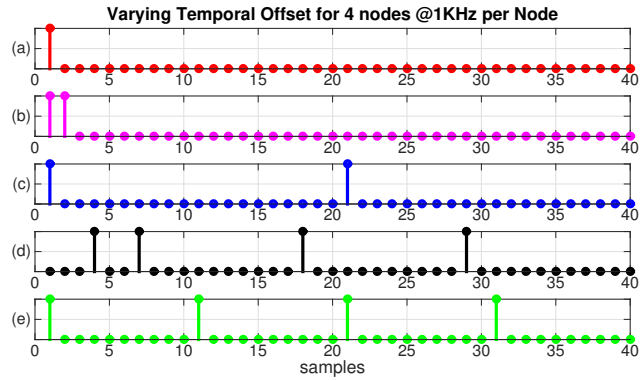


Figure 6.17: Varying temporal offsets from worst to best case sample scenarios for four nodes.

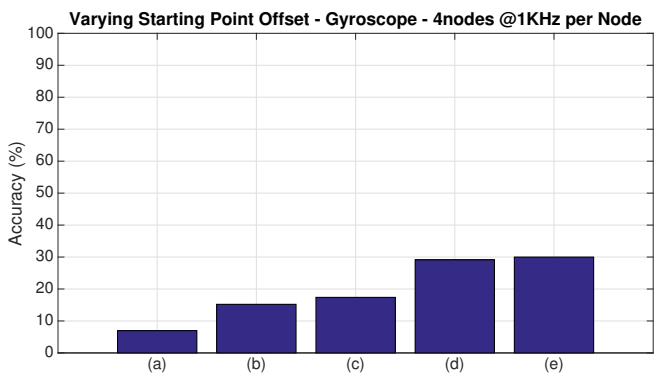


Figure 6.18: Comparison of recognition accuracy of four gyroscopes (F_s=1KHz each) sampled at different temporal offset.

the estimation depends on many factors including the frequency of the signal to sample, the number of nodes sampling, and the sampling frequency per node. This idea is illustrated in Figure 6.20.

Figure 6.20(a) displays a scenario where different sensors sample a sine wave which varies its frequency from 2 Hz to 10 Hz), with a constant sampling frequency of 25 Hz across the sensor nodes (Sensor 1, 2, and 3). The starting point of the nodes were not synchronized and were taken at random.

We demonstrate that even without synchronizing the nodes, the attacker gains enough information to estimate a sensible signal for certain portion of the original signal. Figure 6.20(b) depicts this idea, where the solid line shows the estimated signal after interleaving the sampled data from the sensors. Specifically, the 2 Hz portion of the sine wave can be estimated more closely than that of the 10 Hz portion, even though the three sensors did not sample with

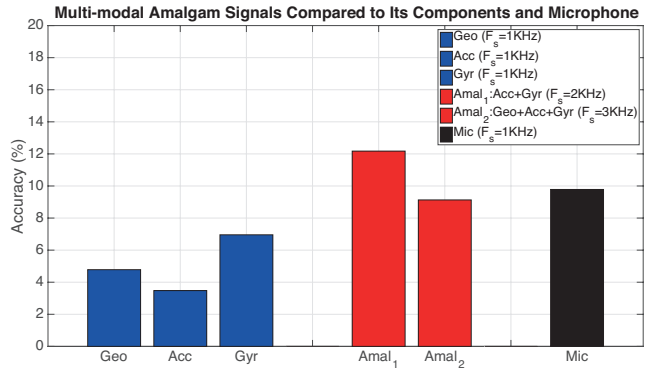
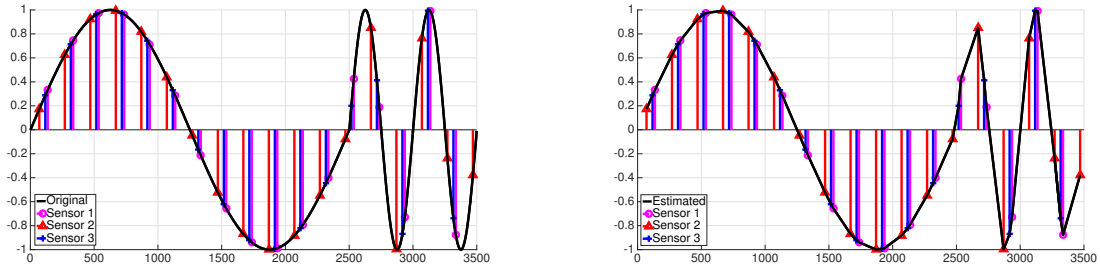


Figure 6.19: Comparison of two multi-modal *Amalgam* signals: (1) Acc+Gyr and (2) Geo+Acc+Gyr each with 1KHz versus their components and microphone as baseline (e.g., Geo, Acc, Gyr, and Mic at 1KHz each).



(a) Original signal sampled by sensors 1, 2, and 3

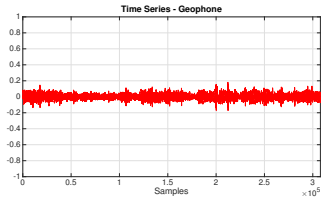
(b) Estimated signal from the sampled data

Figure 6.20: An example of Distributed TI-ADC and its effects when sensors 1, 2, and 3 are sampling the original signal with random temporal offset.

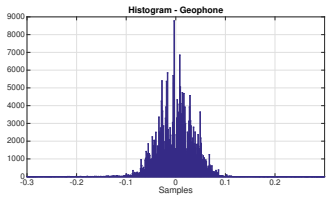
an evenly distributed temporal offset. This is intuitive as more points are sampled for the slower portion of the signal. However, the 10 Hz portion of the signal is not well estimated as shown from the same figure.

6.5.3.3 *Amalgam* Signal Simulation

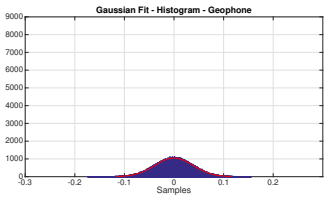
A realistic simulation of sensor node requires acknowledgment of the noise that are unique to each physical sensors. Using a Gaussian fit, we make an assumption that sensors of the same sensor modality has similar signal to noise ratio, and therefore, Gaussian noise of similar variance. In the aforementioned experiments, we sample ambient noise (in a quiet room) from each sensor to estimate the inherent noise distribution in each sensor modality. The values that are sampled are interpreted as a result of a Gaussian noise corrupting the audio signal.



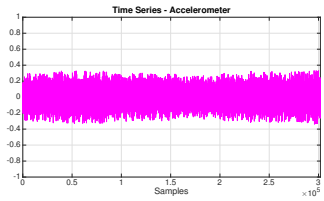
(a) Inherent noise - Geophone



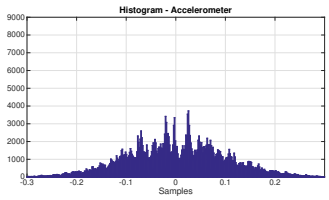
(b) Histogram - Geophone



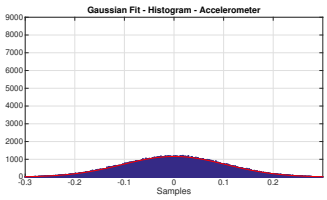
(c) Gaussian Fit - Geophone



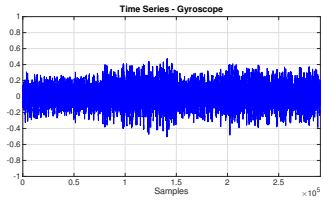
(d) Inherent noise - Accelerometer



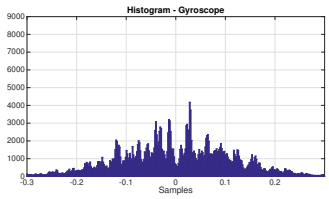
(e) Histogram - Accelerometer



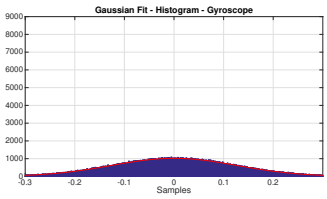
(f) Gaussian Fit - Accelerometer



(g) Inherent noise - Gyroscope



(h) Histogram - Gyroscope



(i) Gaussian Fit - Gyroscope

Figure 6.21: Inherent noise in time series of geophone, accelerometer, and gyroscope and the corresponding histogram and Gaussian fit. The time series data are collected from a quiet room.

We create a generative model to model the noise characteristic of each sensor modality and then estimate the Gaussian fit of such profile. This profile is then used to create multiple instances of possible noise given a sensor. As we add this known noise to the signals we acquired, we simulate realistic sensor data. This process is repeated for all signals used in the present study. Figure 6.21 depicts this process.

6.5.3.4 Multi-modal *Amalgam* Construction

The focus of *PitchIn* is on what the attacker gains when fusing signals from different homogeneous sensors (e.g., fusing distinct geophone signals). We, however, explore the possibilities of fusing multi-modal sensor data to hint at the possibility of recovering speech when only a number of heterogeneous sensor data are available to the attacker. Though more work is required to make this a practical attack with high accuracy, we hope the preliminary evaluation we present in this subsection will demonstrate the feasibility of an even more powerful attack than the aforementioned homogeneous sensor fusion attack.

We compare two multi-modal *Amalgam* signals constructed from fusing each of the following sensors sampling at 1 KHz. $Amal_1$ is composed of an accelerometer and a gyroscope, yielding $F_{s_{Amal}}$=2 KHz. $Amal_2$ is composed of a geophone, an accelerometer, and a gyroscope, yielding $F_{s_{Amal}}$=3 KHz. Figure 6.19 depicts the recognition accuracy of these signals to its components, namely geophone, accelerometer, and gyroscope signals each sampled at 1 KHz. The figure also depicts microphone sampled at 1 KHz for comparison. While the recognition accuracy is low for both *Amalgam* signals (12% and 9%, respectively), we still find that there is an increase in the recognition accuracy compared to their components. $Amal_1$ resulted in a higher accuracy compared to microphone, while $Amal_2$ was comparable. The results hint that sensor fusion across multi-modal sensor data helps to increase the intelligibility.

We note that the accuracy of $Amal_1$ was higher than $Amal_2$, even though the latter included geophone signal. One possible explanation could be that accelerometer and gyroscope had similar signal to noise ratio (SNR) compared to geophone, and interleaving them with the current approach actually yielded more noise in the signal.

	Comparison Pair (F_s)		**p-value**			
			Geo	**Acc**	**Gyr**	**Mic**
Figure 6.11	1KHz	2KHz	.86	<0.001	.41	<.001
	2KHz	4KHz	<.001	<.001	<.001	<.001
	4KHz	8KHz	<.001	<.001	<.001	<.001
	1KHz	8KHz	<.001	<.001	<.001	<.001
Figure 6.13	2KHz	4KHz	<.001	<.001	.43	N/A
	4KHz	8KHz	.24	<.001	.43	N/A
	2KHz	8KHz	<.001	<.001	.22	N/A
Figure 6.14	2KHz	4KHz	<.001	<.001	.05	N/A
	4KHz	8KHz	<.001	<.001	.28	N/A
	2KHz	8KHz	<.001	<.001	<.001	N/A

Table 6.3: Paired t-test for Figures 6.11, 6.13, 6.14 data

6.6 Related Work

We now present related work relevant to *PitchIn*. We first present papers that exploit a non-acoustic sensors to capture sound signals. We then present related work exploring methods to leak side-channel information via sensor data.

Comparison Pair		p-value
Pattern (a)	Pattern (b)	.006
Pattern (c)	Pattern (b)	.52
Pattern (e)	Pattern (b)	<.001
Pattern (a)	Pattern (d)	<.001
Pattern (c)	Pattern (d)	.003
Pattern (e)	Pattern (d)	.84

Table 6.4: Paired t-test for Figure 6.18 data

Comparison Pair (F_s)			p-value
Geophone	1KHz	3KHz	.006
Accelerometer	1KHz	3KHz	.012
Gyroscope	1KHz	3KHz	.41
Geophone	1KHz	2KHz	.005
Accelerometer	1KHz	2KHz	<.001
Gyroscope	1KHz	2KHz	.06

Table 6.5: Paired t-test for Figure 6.19 data

6.6.1 Sensors Capturing Acoustic Signals

Sensors in Smartphones. Recent research has demonstrated keyword detection using an accelerometer [142] and a gyroscope [97] in smartphones. Gyrophone demonstrates that commercial gyroscopes that are implemented in smartphones are capable of capturing acoustic signal even at low sampling frequency [97]. With proper signal processing and machine learning algorithms, this is enough to show speaker identification and speech finger printing. AccelWord demonstrates hot word detection using accelerometer, while achieving low energy consumption [142]. In addition to demonstrating high accuracy in hot word detection, this work also demonstrates the feasibility of an accelerometer capturing rich data more so than conventionally expected.

However, both of these approaches rely on machine learning to train a classifier on a small, predefined group of keyword fingerprints (on the order of tens of words) and later test whether the spoken words' fingerprints match the trained fingerprints, neither reconstructing intelligible speech signals. While these are promising first steps, each work mainly focuses on recovering fingerprints of a small predefined word group. Furthermore, we find that Gyrophone is limited as a practical eavesdropping tool because of the low recognition accuracies when evaluating *speaker-independent* experiments, which resembles a

more realistic attack scenario than *speaker-dependent* expeirment, yet only yielding 7% to 17% on different phones. Gyrophone also provides a preliminary evaluation of merging two gyroscope readings from different smartphones to increase the overall sampling frequency. While Gyrophone neglects to evaluate the results of *speaker-independent* experiments of interleaving two gyroscope signals, we imply that the results must be less than 17% because accuracies of interleaved signals cannot be higher than that of a single sensor.

In this thesis, we are rather more interested in focusing on reconstructing intelligible speech signals *without restriction of predefined keywords nor any prior training*. Instead of predefined keywords, we can leverage any additional context information relevant to the deployment scenario to infer a restricted language model that is independent of the *Amalgam* signal, which aids in speech intelligibility. Hence, the problem we are tackling is necessarily more challenging than the previous approaches because there is no prior restriction on possible fingerprints when the *Amalgam* signal is constructed, requiring much more information to be extracted from the *Amalgam* signal.

Sensors embedded in Non-smartphone Devices. There have been approaches to capture acoustic signals from non-smartphone environments as well. Son et al. describe how gyroscopes respond to acoustic signals of certain frequency, enough to malfunction the flight control of drones [124]. Visual Microphone leverages a camera to capture small vibrations on object surfaces due to sound waves, which recovers the acoustic signal of the sound source [45]. Once again, while *PitchIn* has a synonymous initial idea of capturing sound signals from non-acoustic sensors, we are more interested in fusing disparate non-acoustic sensors that inherently are sampled at low sampling frequencies.

6.6.2 Side-Channel Attacks

ACComplice presents a side-channel attack on an accelerometer in a smartphone by inferring a driver's starting location within a 200 meter radius, along with the traveled route [69]. ACCessory also exploits vulnerabilities of an accelerometer in a smartphone by inferring tapped keystrokes, and is able to extract six character passwords within a median of 4.5 trials [105]. spiPhone uses accelerometer readings of a smartphone placed close to a computer keyboard to infer text entered on the keyboard [94]. These work look into exploiting sensor side-channel vulnerabilities from a single device. *PitchIn*, however, looks into interesting potential vulnerabilities when fusing sensor signals from different devices.

6.7 Discussion

This section presents practical considerations of *PitchIn*.

6.7.1 Time Synchronization

PitchIn requires the devices to be tightly synchronized. Otherwise, it is difficult for an attacker to fuse the aggregated sensor data collected from the network simply because the timestamps from each sensor data are not correlated. However, we are inspired by previous work in time interleaving ADCs (of local devices) that make use of a known reference signal to try to detect and correct timing mismatches or skews among signals sampled by different ADCs. While it is infeasible for an attacker in *PitchIn* to have such a reference signal, we claim that it is feasible for an attacker to perform a manual search (in a bruteforce manner) to shift and find optimal results. While this may be time consuming, it is certainly feasible due to the nature of offline attacks.

Furthermore, similar to how synchronization using NTP is common today, we carefully speculate that a more accurate time synchronization protocols such as Precision Time Protocol (PTP) may be prevalently used in the near future among the IoT devices, as we already find many open source libraries that support PTP protocol on even cheap devices like Arduino [3]. PTP is sufficiently accurate to aid the attacker because *PitchIn* devices require sampling frequencies far less than 8 KHz per node, which translates to a minimum of 125 microsecond per sample, well above the sub-microsecond synchronization accuracy range of PTP.

6.7.2 Amplification

As mentioned in Section 6.5.1, the sensor output were amplified in hardware using operational amplifiers (op-amps) before being interfaced to the Arduino's ADC. We note, however, that the hardware amplification reflects reality as many IoT devices are manufactured with circuitry that leverages hardware amplifiers for sensors [1]. In addition, many IoT devices use digital MEMS sensors, which already come equipped with op-amps within the MEMS circuitry [5].

6.7.3 Automating the Attack

An attacker may automate *PitchIn* attack by feeding in the results obtained by *PitchIn* to an existing Automatic Speech Recognition (ASR) engine. While we had conducted a preliminary experimentation with publicly available Speech Recognition Engine [140], the results were not satisfying, due to the fact that the ASR is trained with microphone data. From consultations with speech recognition experts, we are hopeful that if an attacker trains an ASR with non-acoustic sensors with varying sampling rate, it would most likely yield a relatively high accuracies.

6.8 Chapter Summary

We present *PitchIn* to demonstrate a feasibility of fusing non-acoustic sensors (e.g., geophone, accelerometer, gyroscope) to reconstruct intelligible speech signals using various speech processing techniques. *PitchIn* minimizes per-node sampling frequency by leveraging a distributed Time Interleaved Analog-Digital-Converter (TI-ADC) across network of sensor devices. We conduct user studies to evaluate the intelligibility of the reconstructed signals. *PitchIn* achieves speech recognition accuracy ranging from 79% to 35% depending on the sensor modalities, sampling rate, and number of nodes. We find many potential extensions to *PitchIn*, including increasing scalability of *PitchIn* attack by leveraging automated speech recognition engines to create a fully automated remote eavesdropping tool.

Chapter 7

Summary of Contributions and Future Directions

7.1 Summary of Contributions

This thesis is motivated by the security challenges arising from transformation of computing paradigm from cyber domain to cyber-physical domain. Specifically, traditional security approaches are limited to protecting data with regard to *identities*. For example, authentication is verification of identity. However, in the physical world, it is equally important to consider the physical relationships between the interacting parties. Hence, I present analysis on solutions for IoT devices to bind *identities* and their *physical relationships* together using sensor data (i.e., Signals-of-Opportunity) to measure their relative physical context. More specifically, I present the solution of allowing IoT devices to prove their unique set of relative physical context, which is governed by how the devices are constrained in terms of physical boundary. As exemplary scenarios, I investigated secure pairing of IoT devices in varying application scenarios with different levels of physical constraints including in-vehicle environment, smart home, and semi-autonomous vehicles. I summarize the contributions and key finding of this thesis as following.

In Chapter 3, I demonstrate how devices perform secure pairing when they are provided with the *most constrained* physical environment. Specifically, a car's glove compartment acts as a tightly managed physical boundary, ensuring that the attackers outside of the compartment cannot access any messages transmitted inside encoded in light pulses. Through this chapter, I demonstrate how a driver or passenger can establish a secure connection

between a car's infotainment system and his/her smartphone while preserving usability and low hardware or deployment cost.

In Chapter 4, I demonstrate how devices perform secure pairing when the physical constraint is relaxed. Specifically, a single detached house acts as a physical boundary which attenuates much of the signals so that the attacker's device outside has low fidelity of information. Through this chapter, I also demonstrate how I can enable context-based pairing with dissimilar sensor types by utilizing common timing information as sources of common entropy to be input to a fuzzy commitment protocol.

In Chapter 5, I demonstrate how devices perform secure pairing in a least constrained environment. Specifically, I study a truck platooning on a road, where there are no clear physical boundary to restrict attackers as opposed to the previous chapters. However, I demonstrate that the trucks driving on the same lane would experience similar road characteristics, which unintended vehicles on the adjacent lanes cannot experience, thereby using the bumps on the road as common source of entropy in the cryptographic key establishment protocols.

While I demonstrate how sensed data can complement cyber-physical security in the above chapters, I also demonstrate that sensing may pose security vulnerabilities to IoT and CPS in Chapter 6. Specifically, I investigate the feasibility of launching a side-channel (eavesdropping) attack by reconstructing intelligible speech signals, by fusing multiple non-acoustic IMU-based sensor data available in a smart home environment. Even though each individual sensing device samples at a low rate, I interleaved the signals across sensors to yield a reconstructed signal that is intelligible due to high overall sampling rate. Through this work, I demonstrate the importance of shifting our defense strategies from exploits on individual devices to coordination of devices. This is because I find that a large portion of the research community has concentrated on finding and defending against vulnerabilities of individual sensors or devices, and what the posed risks are for the users. However, I believe that it is also important to explore new vulnerabilities if an attacker compromises data collected from multiple devices located within the same environment.

7.2 Future Directions

In the future, I plan to improve the limitations of my current work. I also plan to continue to identify newer security challenges and investigate solutions in such emerging CPS applications. In addition, I plan to continue to explore newer side-channel attacks exploiting signals-

of-opportunity, and extend my work in non-security oriented problems that contribute to value-added services for cyber-physical systems.

7.2.1 Improving Limitations of Current Work

I plan to extend my thesis by improving the limitations of the current work presented in the aforementioned chapters. Specifically, I plan to (1) understand the physical phenomena by characterizing and modeling the Signals-of-Opportunity; and (2) measure and quantify how much entropy can be extracted from the characterized physical phenomena.

Characterize and Model Physical Phenomena: Investigating and modeling characteristics of various corresponding physical phenomena will be extremely helpful to fully understanding the limitations of my work and simultaneously understanding the capabilities of the attacker. For example, in the current version of Chapter 4, we define a notion of an "attenuation factor" (discussed in Chapter 4.2 and 4.7) due to the wall of a house, which provides some insulation of signals to propagate to the outside. While we demonstrate empirically that the attackers do not observe with sufficient fidelity of information, it would be more desirable to model this attenuation factor for varying types of signals. For example, understanding and modeling how different structures limits the penetration of different types of signals induced from different activities inside the house will directly lead to quantifying the attacker's capabilities. Specifically, I anticipate to apply similar notion to Wyner's "Wire-tap Channel" [139] to the physical sensing scenario, where I can treat the transmitter, receiver, and noiseless communication channel as the event source, sensor, and the physical medium within the physical boundary, respectively. Furthermore, I can treat the attacker's degraded wire-tapping channel as the physical boundary with certain attenuation factor.

Measure and Quantify Extracted Entropy: Taking the aforementioned characterized physical phenomena, I also plan to measure and quantify the strength of the randomness of corresponding Signals-of-Opportunity (i.e., modeling the entropy). While we are currently utilizing Fuzzy Commitment Schemes [76] to utilize random activities from the environments, the current work is limited in quantifying how many bits of entropy can be extracted that are close to being uniformly distributed. I will further investigate the use of randomness extractor schemes [4, 51, 127] to measure, model, and quantify random activities as sources of entropy. Furthermore, to enhance the entropy bits, I will also investigate solutions to "inject" entropy to the physical world. This approach would help the devices to extract shared entropy by deliberately injecting randomness to the devices within the physical boundary. This may be

realized by introducing a *signal injecting device* synonymous to a physical pseudo-random number generator (e.g., device with vibration motor or speaker) that outputs signals such as vibration or sound that are encoded random bits. This is analogous to traditional key establishment schemes that provide "deliberate entropy" [29].

7.2.2 Exploring Security Challenges in Emerging Applications

Newer application domains such as smart buildings, vehicles, and cities inevitably introduce unforeseen challenges with regard to cross-platform verification and coordination. For example, I envision applications of autonomous cars exchanging messages for driving safety. Although authenticating the message and the sender is an important problem, prioritizing the message with regard to the relative location is equally important. Context verification would be a potential solution to establish trust across cars on a road in an ad-hoc manner, as well as for verifying relative locations. In fact, I have engaged in ongoing and past collaborations with research labs at automotive companies such as Ford and General Motors to tackle such ideas, some of which have resulted in patent publication [82].

7.2.3 Side-Channel Attacks Exploiting Contextual Information

I plan to continue my investigation of unexplored side-channel threats that may be exploited from contextual information. This will be a natural extension of my past work on side-channel attacks to infer a victim's driving locations and keystrokes based on signals from an accelerometer in a smartphone [69, 105], as well as the aforementioned eavesdropping of speech signals from non-acoustic IMU-based sensors [66] in Chapter 6. Investigating new side-channel attacks is important because it allows me to gain insight into the system, which can often be used to (1) turn a vulnerability into a functionality (e.g., turning an eavesdropping attack into speech recognition), as there is a fine line between risk and value-added services. It can also be used to (2) design defense mechanisms to protect against such vulnerabilities.

7.2.4 Non-Security Oriented Value-Added Services for IoT and CPS

I am also interested in extending my work to solve non-security oriented problems. For example, GPS in urban areas is known to be error-prone and has limitations in providing fine-grained navigation services (e.g., lane-level granularity). I am currently investigating the feasibility of verifying physical context from road characteristics (e.g., cracks, bumps, and

patches) using a car's accelerometer readings to augment the noisy GPS readings [67].

Furthermore, I plan to utilize contextual information to investigate solutions to problems such as time synchronization without requiring specialized hardware. For instance, I have been exploring the possibility of leveraging commonly perceived sound and vibration as sources of synchronization.

Bibliography

[1] ADXL103/ADXL203 Datasheet. Analog Devices Data Sheet. Cited on page 106.

[2] Global Markets and Technologies for Sensors. http://www.bccresearch.com/market-research/instrumentation-and-sensors/sensors-ias006f.html. Cited on page 81.

[3] PTPd. http://ptpd.sourceforge.net/. Cited on page 106.

[4] Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, July 2001. Cited on page 111.

[5] LIS331DLH: MEMS digital output motion sensor ultra low-power high performance 3-axes "nano" accelerometer. STMicroelectronics – Datasheet, 2009. Cited on page 106.

[6] ADC081000,ADC08D1000: Interleaving ADCs for Higher Sample Rates. Texas Instruments Whitepaper SNAA111, 2011. Cited on pages 84 and 87.

[7] ADX - Time-interleaving Technology for Analog-to-Digital Converters (ADCs). http://spdevices.com/index.php/interleaving, 2015. Cited on pages 84 and 87.

[8] Arduino Ethernet Shield. https://www.arduino.cc/en/Main/ArduinoEthernetShield, 2015. Cited on page 94.

[9] Arduino FAT16/FAT32 Library. https://github.com/greiman/SdFat, 2015. Cited on page 94.

[10] Arduino/Genuino UNO. https://www.arduino.cc/en/Main/arduinoBoardUno, 2015. Cited on pages 49, 75, and 92.

[11] Adaptive Cruise Control . https://goo.gl/Jbc7dv, 2016. Cited on page 67.

[12] Daimler Trucks is Connecting Its Trucks With The Internet. http://media.daimler.com/marsMediaSite/en/instance/ko/Daimler-Trucks-is-connecting-its-trucks-with-the-internet.xhtml?oid=

9920445, 2016. Cited on page 65.

[13] ElectricImp. https://electricimp.com/, 2016. Cited on page 36.

[14] ElectricImp BlinkUp™. https://electricimp.com/platform/blinkup/, 2016. Cited on page 36.

[15] European Truck Platooning Challenge – Creating Next Generation Mobility. https://www.eutruckplatooning.com/home/default.aspx, 2016. Cited on page 65.

[16] Impact of Platooning on Traffic Efficiency. https://trid.trb.org/view.aspx?id=1262546, 2016. Cited on pages 65 and 68.

[17] Peloton. http://peloton-tech.com/, 2016. Cited on page 65.

[18] Platooning. https://www.volvogroup.com/en-en/news/2018/feb/truck-platooning-on-european-roads.html, 2016. Cited on page 65.

[19] Wireless SD Shield. https://www.arduino.cc/en/Main/ArduinoWirelessShield, 2016. Cited on page 49.

[20] Spencer Ackerman and James Ball. Optic Nerve: millions of Yahoo webcam images intercepted by GCHQ. http://www.theguardian.com/world/2014/feb/27/gchq-nsa-webcam-images-internet-yahoo, 2014. Cited on pages 83 and 89.

[21] Knowles Acoustics. MD9745APA-1 Product Specification. http://media.digikey.com/pdf/Data%20Sheets/Knowles%20Acoustics%20PDFs/MD9745APA-1.pdf, 2003. Cited on page 48.

[22] W.F. Alliance. Wi-fi protected access: Strong, standards-based, interoperable security for today's wi-fi networks. *Retrieved March*, 1:2004, 2003. Cited on page 31.

[23] Amazon. *Amazon Echo*. https://www.amazon.com/Amazon-Echo-Bluetooth-Speaker-with-WiFi-Alexa/dp/B00X4WHP5E. Cited on page 38.

[24] Amazon. Amazon Mechanical Truk. https://www.mturk.com/mturk/welcome, 2015. Cited on page 94.

[25] Tali Arbel. Verizon business accounts hacked. https://www.usnews.com/news/articles/2016-03-25/verizon-business-accounts-hacked, 2016. Cited on pages 83 and 89.

[26] Atmel. Atmel 8-bit AVR Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash. https://www.sparkfun.com/datasheets/Components/SMD/

ATMega328.pdf. Cited on page 93.

[27] Audacity. DC Offset. http://manual.audacityteam.org/o/man/dc_offset.html. Cited on page 91.

[28] Lorraine E Bahrick, Robert Lickliter, and Ross Flom. Intersensory redundancy guides the development of selective attention, perception, and cognition in infancy. *Current Directions in Psychological Science*, 13(3):99–102, 2004. Cited on page 38.

[29] Dirk Balfanz, Dirk Balfanz, Glenn Durfee, Glenn Durfee, Rebecca E. Grinter, Rebecca E. Grinter, D. K. Smetters, D. K. Smetters, Paul Stewart, and Paul Stewart. Network-in-a-box: How to set up a secure wireless network in under a minute. In *Usenix Security*, 2004. Cited on page 112.

[30] Dirk Balfanz, Diana K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *NDSS*, 2002. Cited on page 15.

[31] Elaine Barker. Recommendation for Key Management. NIST Special Publication 800-57, 2016. Cited on page 57.

[32] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*, 1992. Cited on page 20.

[33] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. The Keccak sponge function family. http://keccak.noekeon.org/, 2012. Cited on page 57.

[34] Norbert Bissmeyer, Joël Njeukam, Jonathan Petit, and Kpatcha M. Bayarou. Central misbehavior evaluation for vanets based on mobility data plausibility. In *Proceedings of the Ninth ACM VANET*, 2012. Cited on page 66.

[35] Bosch. ISC-BPR2 - Blue Line Gen2 PIR Motion Detectors. http://resource.boschsecurity.com/documents/BlueLine_Gen_2_Data_sheet_enUS_2603228171.pdf. Cited on page 36.

[36] U.S. Census Bureau. *Average number of people per household in the United States from 1960 to 2016*, 2017. https://www.statista.com/statistics/183648/average-size-of-households-in-the-us/. Cited on page 60.

[37] S Butterworth. On the theory of filter amplifiers. In *Experimental Wireless and the Wireless Engineer*, 1930. Cited on page 92.

[38] IO Sensor Nederland b.v. SM-24 Geophone Element. https://goo.gl/2hG6xG. Cited

on pages 48, 85, and 93.

[39] M. Cagalj, S. Capkun, and J. P. Hubaux. Key agreement in peer-to-peer wireless networks. *Proceedings of the IEEE*, 94(2):467–478, Feb 2006. Cited on page 13.

[40] Eric Cheng and Paul Hudak. Audio Processing and Sound Synthesis in Haskell. January 2009. Cited on page 24.

[41] Car Connectivity Consortium. Mirror Link. http://www.mirrorlink.com/. Cited on page 16.

[42] R. Crochiere and Rabiner L. Multirate Digital Signal Processing, 1983. Cited on pages 84 and 88.

[43] Cuisinart. *SPB-650 Blender*. http://www.cuisinart.com/products/blenders/spb-650.html. Cited on page 51.

[44] Soodeh Dadras, Ryan M. Gerdes, and Rajnikant Sharma. Vehicular platooning in an adversarial environment. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '15, pages 167–178, New York, NY, USA, 2015. ACM. Cited on page 78.

[45] Abe Davis, Michael Rubinstein, Neal Wadhwa, Gautham J. Mysore, Frédo Durand, and William T. Freeman. The visual microphone: Passive recovery of sound from video. *ACM Trans. Graph.*, 33(4):79:1–79:10, July 2014. Cited on page 105.

[46] Bruce DeBruhl, Sean Weerakkody, Bruno Sinopoli, and Patrick Tague. Is your commute driving you crazy?: A study of misbehavior in vehicular platoons. In *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, WiSec '15, pages 22:1–22:11, New York, NY, USA, 2015. ACM. Cited on pages 78 and 79.

[47] Deloitte. The Digital Predictions 2015. The Deloitte Consumer Review, 2015. Cited on page 35.

[48] Analog Devices. ADXL330 Datasheet. https://www.sparkfun.com/datasheets/Components/ADXL330_0.pdf. Cited on page 85.

[49] Analog Devices. ADXL335 Datasheet. http://www.analog.com/media/en/technical-documentation/data-sheets/ADXL335.pdf. Cited on pages 48, 75, 85, and 93.

[50] Whitefield Diffie and Martin E. Hellman. New directions in cryptography. In *IEEE Trans. Information. Theory*, volume IT-22, pages 644–654, Nov 2009. Cited on page

45.

[51] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.* Cited on pages 43 and 111.

[52] Ecolink. *Z-Wave Motion Detector with Pet Immunity*. http://www.discoverecolink.com/product/pirzwave2-eco/. Cited on page 38.

[53] Eero. *An Entirely New Approach to Home WiFi*. https://eero.com/. Cited on page 62.

[54] LG Electronics. Magic Remote. http://webostv.developer.lge.com/api/webos-service-api/magic-remote/?wos_flag=getsensordata. Cited on pages 82 and 87.

[55] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. The pothole patrol: Using a mobile sensor network for road surface monitoring. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, MobiSys '08, pages 29–39, New York, NY, USA, 2008. ACM. Cited on page 79.

[56] Fibaro. *Motion Sensor*. http://www.fibaro.com/us/the-fibaro-system/motion-sensor. Cited on page 38.

[57] Center for Advanced Studies in Adaptive Systems. *WSU CASAS Datasets*. http://casas.wsu.edu/datasets/. Cited on page 59.

[58] Christian Gehrmann, Chris J. Mitchell, and Kaisa Nyberg. Manual authentication for wireless devices. In *RSA Cryptobytes*, 2004. Cited on page 32.

[59] D.V. Giri and F.M. Tesche. *Modeling of Propagation Losses in Common Residential and Commercial Building Walls*, 2013. Interaction Note 624. Cited on page 40.

[60] G. Goertzel. An algorithm for the evaluation of finite trigonometric series. *American Mathematical Monthly*, 65:34 – 35, 1958. Cited on page 24.

[61] Google. *Google Home*. https://home.google.com/. Cited on page 38.

[62] Google. *Google WiFi: Home Wi-Fi, simply solved*. https://madeby.google.com/wifi/. Cited on page 62.

[63] Chuck Goudie and Ross Weidner. Home hackers: Digital invaders a threat to your house. http://abc7chicago.com/technology/

home-hackers-digital-invaders-a-threat-to-your-house/515520/, 2015. Cited on pages 83 and 89.

[64] Bluetooth Core Specification Working Group. Bluetooth simple pairing Whitepaper. Bluetooth SIG Whitepaper '06. Cited on page 31.

[65] Thread Group. *What is Thread?* http://threadgroup.org/What-is-Thread/Overview. Cited on page 61.

[66] Jun Han, Albert Jin Chung, and Patrick Tague. Pitchin: Eavesdropping via intelligible speech reconstruction using non-acoustic sensor fusion. In *16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2017. Cited on pages 81 and 112.

[67] Jun Han, Madhumitha Harishankar, Shi Su, Shijia Pan, Hae Young Noh, Pei Zhang, Marco Gruteser, and Patrick Tague. "AcceLane: Lane-level localization of vehicles using road condition monitoring". Pending Submission. Cited on page 113.

[68] Jun Han, Madhumitha Harishankar, Xiao Wang, Albert Jin Chung, and Patrick Tague. Convoy: Physical context verification for vehicle platoon admission. In *18th International Workshop on Mobile Computing Systems and Applications (HotMobile)*, Feb 2017. Cited on pages 43 and 60.

[69] Jun Han, Emmanuel Owusu, Le T. Nguyen, Adrian Perrig, and Joy Zhang. Accomplice: Location inference using accelerometers on smartphones. In *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, pages 1–9, Jan 2012. Cited on pages 81, 105, and 112.

[70] Jun Han, Abhishek Shah, Mark Luk, and Adrian Perrig. Don't sweat your privacy using humidity to detect human presence. In *Workshop On UbiComp Privacy - Technologies, Users, Policy (UbiPriv)*, 2007. Cited on page 35.

[71] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006. Cited on page 58.

[72] Insteon. *Low Voltage/Contact Closure Interface.* http://www.insteon.com/io-module. Cited on page 38.

[73] Texas Instruments. LMV3xx Low-Voltage Rail-to-Rail Output Operational Amplifiers. http://www.ti.com/lit/ds/symlink/lmv324.pdf, 2014. Cited on page 93.

[74] P3 International. Kill-A-Watt P4400. http://www.p3international.com/products/

p4400.html. Cited on pages 38 and 48.

[75] Larry P. Jedele. *Energy Attenuation Relationships from Construction Vibrations*, 1985. Vibration Problems in Geotechnical Engineering. ASCE Convention in Detroit, Michigan. Cited on page 40.

[76] A. Juels and M. Sudan. A fuzzy vault scheme. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, pages 408–, 2002. Cited on pages 43, 67, 70, and 111.

[77] R. Kainda, I. Flechais, and AW Roscoe. Usability and security of out-of-band channels in secure device pairing protocols. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, page 11. ACM, 2009. Cited on page 32.

[78] J.B. Kenney. Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, July 2011. Cited on page 65.

[79] David J Ketchen and Christopher L Shook. The application of cluster analysis in strategic management research: an analysis and critique. *Strategic management journal*, 17(6):441–458, 1996. Cited on page 47.

[80] Jungsuk Kim and Ian Lane. Accelerating large vocabulary continuous speech recognition on heterogeneous cpu-gpu platforms. In *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*, pages 3291–3295, May 2014. Cited on page 89.

[81] Sukun Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pages 254–263, April 2007. Cited on pages 82 and 87.

[82] Yu Seung Kim, Jun Han, and Patrick Tague. "Inter-vehicle authentication using visual contextual information". In *Patent US20170132477 A1*, 2017. Cited on page 112.

[83] A. Kumar, N. Saxena, G. Tsudik, and E. Uzun. A comparative study of secure device pairing methods. *Pervasive and Mobile Computing*, 5(6):734–749, 2009. Cited on page 32.

[84] Cynthia Kuo, Mark Luk, Rohit Negi, and Adrian Perrig. Message-in-a-bottle: User-friendly and secure key deployment for sensor nodes. In *ACM Conference on Embedded Networked Sensor System (SenSys)*, Nov 2007. Cited on page 13.

[85] Cynthia Kuo, Jesse Walker, and Adrian Perrig. Low-cost manufacturing, usability, and

security: An analysis of bluetooth simple pairing and wi-fi protected setup. In *USEC*, 2007. Cited on pages 16 and 31.

[86] Wise Lab. Events 2.0 Technical Guide. http://wise.ece.cmu.edu/redmine/projects/firefly/wiki/FireFly3_x2. Cited on pages 82 and 87.

[87] Michael P. Lammert, Adam Duran, Jeremy Diez, Kevin Burton, and Alex Nicholson. Effect of Platooning on Fuel Consumption of Class 8 Vehicles Over a Range of Speeds, Following Distances, and Mass. SAE Technical Report, 2014. Cited on page 65.

[88] Sven Laur, N. Asokan, and Kaisa Nyberg. Efficient mutual data authentication using manuallyauthenticated strings. In *Cryptography and Network Security*, pages 90–107, 2006. Cited on page 21.

[89] Christine Laurendeau and Michel Barbeau. *Ad-Hoc, Mobile, and Wireless Networks: 5th International Conference, ADHOC-NOW 2006, Ottawa, Canada, August 17-19, 2006. Proceedings*, chapter Threats to Security in DSRC/WAVE, pages 266–279. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. Cited on page 66.

[90] Yunxin (Jeff) Li. *Quality, Reliability, Security and Robustness in Heterogeneous Networks: 7th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, QShine 2010, and Dedicated Short Range Communications Workshop, DSRC 2010, Houston, TX, USA, November 17-19, 2010, Revised Selected Papers*, chapter An Overview of the DSRC/WAVE Technology, pages 544–558. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Cited on page 65.

[91] T. Mahmud, M. Hasan, A. Chakraborty, and A. K. Roy-Chowdhury. A poisson process model for activity forecasting. In *IEEE International Conference on Image Processing (ICIP)*, 2016. Cited on page 58.

[92] Rahul Mangharam, Anthony Rowe, and Raj Rajkumar. Firefly: A cross-layer platform for real-time embedded wireless networks. *Real-Time Syst.*, 37(3):183–231, December 2007. Cited on pages 82 and 87.

[93] Chung-Tse Mar, Mat Hans, Mark Smith, Tajana Simunic, and Ronald Schafer. A High-Quality, Energy Optimized, Real-Time Sampling Rate Conversion Library for the StrongARM Microprocessor. Technical Report HPL-2002-159, 2002. Cited on pages 84 and 88.

[94] Philip Marquardt, Arunabh Verma, Henry Carter, and Patrick Traynor. (sp)iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers. In

Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS '11, pages 551–562, New York, NY, USA, 2011. ACM. Cited on pages 81 and 105.

[95] Christopher Mazur. *Physical Characteristics of Housing: 2009–2011*, 2013. American Community Survey Briefs – U.S. Census Bureau, Department of Commerce. Cited on page 41.

[96] Jonathan McCune, Adrian Perrig, and Michael Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2005. Cited on page 32.

[97] Yan Michalevsky, Dan Boneh, and Gabi Nakibly. Gyrophone: Recognizing speech from gyroscope signals. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 1053–1067, San Diego, CA, August 2014. USENIX Association. Cited on pages 83 and 104.

[98] Blue Microphones. *Yeti*. http://www.bluedesigns.com/products/yeti. Cited on page 49.

[99] Markus Miettinen, N. Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and Majid Sobhani. 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, pages 880–891, New York, NY, USA, 2014. ACM. Cited on pages 13, 43, 60, 67, 70, and 79.

[100] Andrés Molina-Markham, Prashant Shenoy, Kevin Fu, Emmanuel Cecchet, and David Irwin. Private memoirs of a smart meter. In *Proceedings of the 2Nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, BuildSys '10, pages 61–66, New York, NY, USA, 2010. ACM. Cited on page 35.

[101] Nestle. *Nespresso Pixie Carmine*. https://www.nespresso.com/us/en/order/machines/original/Nespresso-Pixie-Dark-Red. Cited on page 50.

[102] Nexia. *Schlage Motion Sensor*. http://www.nexiahome.com/compatible-products/schlage-home-motion-sensor/. Cited on page 38.

[103] Steven Norton. Internet of Things Market to Reach $1.7 Trillion by 2020: IDC. http://blogs.wsj.com/cio/2015/06/02/internet-of-things-market-to-reach-1-7-trillion-by-2020-idc/, jun 2015. Cited on page 81.

[104] Notion. *Home awareness, simplified. Monitor your home with a single sensor, wherever you are.* http://getnotion.com/. Cited on page 38.

[105] Emmanuel Owusu, Jun Han, Sauvik Das, Adrian Perrig, and Joy Zhang. Accessory: Password inference using accelerometers on smartphones. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, HotMobile '12, pages 9:1–9:6, New York, NY, USA, 2012. ACM. Cited on pages 81, 105, and 112.

[106] Shijia Pan, Amelie Bonde, Jie Jing, Lin Zhang, Pei Zhang, and Hae Young Noh. Boes: Building occupancy estimation system using sparse ambient vibration monitoring. In *Proc. SPIE 9061*, 2014. Cited on pages 82 and 87.

[107] Shijia Pan, Ningning Wang, Yuqiu Qian, Irem Velibeyoglu, Hae Young Noh, and Pei Zhang. Indoor person identification through footstep induced structural vibration. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, HotMobile '15, pages 81–86, New York, NY, USA, 2015. ACM. Cited on pages 35, 38, 82, and 87.

[108] Panasonic. MP Motion Sensor NaPiOn. https://www3.panasonic.biz/ac/e/control/sensor/human/napion/. Cited on page 48.

[109] S. Pasini and S. Vaudenay. Sas-based authenticated key agreement. In *Theory and Practice of Public-Key Cryptography (PKC)*, 2006. Cited on page 21.

[110] Pery Pearson. Sound Sampling. http://www.hitl.washington.edu/projects/knowledge_base/virtual-worlds/EVE/I.B.3.a.SoundSampling.html. Cited on pages 84, 88, and 89.

[111] Colin Percival. Stronger key derivation via sequential memory-hard functions. 2009. *https://www.tarsnap.com/scrypt/scrypt.pdf*. Cited on page 44.

[112] Plume. *Plume WiFi*. https://www.plumewifi.com/. Cited on page 62.

[113] Tom Robinson and Eric Chan. Operating Platoons On Public Motorways: An Introduction To The SARTRE Platooning Programme. Proceedings of the 17th ITS World Congress, Oct 2010. Cited on page 65.

[114] Masoud Rostami, Ari Juels, and Farinaz Koushanfar. Heart-to-heart (h2h): authentication for implanted medical devices. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, CCS '13, pages 1099–1112, New York, NY, USA, 2013. ACM. Cited on page 60.

[115] J. Salowey, A. Choudhury, and D. McGrew. AES Galois Counter Modes (GCM) Cipher Suites for TLS. RFC, 2008. Cited on page 45.

[116] Karen Scarfone and John Padgette. Guide to bluetooth security. *NIST Special*

Publication, 800:121, 2008. Cited on page 15.

[117] Dominik Schurmann and Stephan Sigg. Secure communication based on ambient audio. *IEEE Transactions on Mobile Computing*, 12(2):358–370, February 2013. Cited on pages 14, 60, 70, and 79.

[118] Freescale Semiconductor. *MMA1270KEG Datasheet*. http://www.nxp.com/docs/en/data-sheet/MMA1270KEG.pdf. Cited on page 49.

[119] C.E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, Jan 1949. Cited on page 88.

[120] Hocheol Shin, Yunmok Son, Youngseok Park, Yujin Kwon, and Yongdae Kim. Sampling race: Bypassing timing-based analog active sensor spoofing detection on analog-digital systems. In *10th USENIX Workshop on Offensive Technologies (WOOT 16)*, 2016. Cited on page 62.

[121] Yasser Shoukry, Paul Martin, Yair Yona, Suhas Diggavi, and Mani Srivastava. Pycra: Physical challenge-response authentication for active sensors under spoofing attacks. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, 2015. Cited on page 62.

[122] SkyPaw. SkyPaw Multi Measures – Decibel. http://www.skypaw.com/apps/multimeasures/, 2015. Cited on pages 31 and 94.

[123] Samsung SmartThings. *Multipurpose Sensor*. https://support.smartthings.com/hc/en-us/articles/213496343. Cited on page 38.

[124] Yunmok Son, Hocheol Shin, Dongkwan Kim, Youngseok Park, Juhwan Noh, Kibum Choi, Jungwoo Choi, and Yongdae Kim. Rocking drones with intentional sound noise on gyroscopic sensors. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 881–896, Washington, D.C., August 2015. USENIX Association. Cited on pages 62 and 105.

[125] C. Soriente, G. Tsudik, and E. Uzun. HAPADEP: human-assisted pure audio device pairing. *Information Security*, pages 385–400, 2008. Cited on page 13.

[126] STMicroelectronics. LPY503AL Datasheet. https://www.sparkfun.com/datasheets/Sensors/IMU/lpy503al.pdf. Cited on pages 85 and 93.

[127] L. Trevisan and S. Vadhan. Extracting randomness from samplable distributions. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00, pages 32–, Washington, DC, USA, 2000. IEEE Computer Society. Cited on page

111.

[128] T. Trippel, O. Weisse, W. Xu, P. Honeyman, and K. Fu. Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks. In *2017 IEEE European Symposium on Security and Privacy (EuroS P)*, 2017. Cited on page 62.

[129] Daniel Turner. Hack: The Nintendo Wii. http://www.technologyreview.com/hack/408183/hack-the-nintendo-wii/, 2007. Cited on pages 82 and 87.

[130] Hasan S. Ulusoy, Erol Kalkan, Jon Peter B. Fletcher, Paul Friberg, W. K. Leith, and Krishna Banga. Design and implementation of a structural health monitoring and alerting system for hospital buildings in the united states. In *Proceedings of the 15th World Conference on Earthquake Engineering*, HotMobile '15, 2012. Cited on pages 38, 82, and 87.

[131] E. Uzun, N. Saxena, and A. Kumar. Pairing devices for social interactions: a comparative usability evaluation. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 2315–2324. ACM, 2011. Cited on page 32.

[132] Ersin Uzun, Kristiina Karvonen, and N. Asokan. Usability analysis of secure pairing methods. In *USEC*, 2007. Cited on page 16.

[133] B. van Arem, C. J. G. van Driel, and R. Visser. The impact of cooperative adaptive cruise control on traffic-flow characteristics. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):429–436, Dec 2006. Cited on pages 67 and 68.

[134] K. J. Van Engen, J. E. Phelps, R. Smiljanic, and B. Chandrasekaran. Enhancing speech intelligibility: interactions among context, modality, speech style, and masker. *Experimental Wireless and the Wireless Engineer*, 2014. Cited on pages 89 and 94.

[135] Serge Vaudenay. Secure communications over insecure channels based on short authenticated strings. In *International Cryptology Conference (CRYPTO)*, 2005. Cited on page 21.

[136] Stefan Viehbock. Brute forcing Wi-Fi Protected Setup. When poor design meets poor implementation. http://sviehb.files.wordpress.com/2011/12/viehboeck_wps.pdf. Cited on page 16.

[137] WaspMote. Events 2.0 Technical Guide. http://www.libelium.com/downloads/documentation/events-sensor-board_2.0.pdf. Cited on pages 82 and 87.

[138] Will Knight. 10-4, Good Computer: Automated System Lets Trucks Convoy as One. MIT Technology Review, May 2014. Cited on pages 65 and 68.

[139] A. D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, Oct 1975. Cited on page 111.

[140] Anthony Zhang. Speech Recognition (version 2.0). https://github.com/Uberi/speech_recognition, 2015. Cited on page 107.

[141] Jie Zhang. A survey on trust management for vanets. In *Proceedings of the 2011 IEEE International Conference on Advanced Information Networking and Applications*, AINA '11, pages 105–112, Washington, DC, USA, 2011. IEEE Computer Society. Cited on page 79.

[142] Li Zhang, Parth H. Pathak, Muchen Wu, Yixin Zhao, and Prasant Mohapatra. Accelword: Energy efficient hotword detection through accelerometer. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '15, pages 301–315, New York, NY, USA, 2015. ACM. Cited on pages 83 and 104.