# DESIGNING CRYPTOGRAPHY SYSTEMS FOR GNSS DATA AND RANGING AUTHENTICATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Jason Anderson

December 2024

# Abstract

The Global Positioning System (GPS) has profoundly changed commercial aviation over the last three decades. Today, GPS (and the broader global system, Global Navigation Satellite System (GNSS)) pervades our everyday lives, including safety-critical aspects ranging from commercial aviation to electrical grid time synchronization. Autonomous vehicles rely on GNSS signals to estimate vehicle position and time. GNSS's open nature and ubiquitous adoption lend a potentially fatal vulnerability. Anyone can broadcast spoof signals to fool a receiver into believing GNSS is safe when, in fact, it is hazardous, thus perpetually casting doubt on GNSS accuracy. Spoofed GNSS signals are an undeniable reality of modern conflict zones and are likely to spread worldwide. What started as fooling Pokemon Go can now affect the safety-critical aspects of our lives. This dissertation explores augmenting GNSS signals with cryptographic methods to establish trust in the signals.

A GNSS engineer can use this dissertation to inform the system and receiver design to mitigate spoofing risks using cryptography. From a GNSS provider perspective, this dissertation covers the elements of cryptographic construction for provable security, efficient bandwidth use, and quick authentication time. The primary authentication structure a GNSS Provider should use is Timed Efficient Stream Loss-tolerant Authentication (TESLA). This dissertation describes leveraging TESLA's features, designing the needed security maintenance structures with standard asymmetric authentication cryptography, and asserting the TESLA time-synchronization requirements. Moreover, TESLA enables ranging authentication via ranging code watermarking without additional data-distribution bandwidth. This dissertation describes how to securely perform watermarking in a way that allows straightforward derivation of the distribution of receiver-measured statistics. From a GNSS receiver perspective, this dissertation covers the receiver processing required to establish trust in the GNSS signal, including computing the security they afford.

Previously, the severely constrained GNSS data bandwidth posed significant challenges to incorporating cryptography, often dictating the product (i.e., the service the GNSS constellation could offer). However, a significant shift occurs with the techniques presented in this dissertation. Whereas before, cryptography dictated the product, the product can now dictate the cryptography.

# Acknowledgements

I stand on the shoulders of giants with the aid of many supportive colleagues, friends, and loved ones.

To my dissertation readers, Professors Grace Gao and Dan Boneh: Thank you for your introductory and elective coursework. It was very interesting and informative and helped me conduct the research that formed this dissertation.

To my colleagues who made OSNMA, SBAS Authentication, and Chimera possible: thank you for providing a strong foundation from which I could build the work of this thesis. Countless times, I have been inspired by your work and your herculean efforts with the broader community and our governments to ensure safe navigation systems against upcoming threats.

To Andrew Neish, thank you for your influential dissertation, grounding my research efforts into something immediately useful, and for your support and mentorship.

To Ignacio Fernandez-Hernandez, Cillian O'Driscoll, Jed Dennis, Joe Rushanan, and Jim Gillis: thank you for listening and engaging with my work.

To Juan Blanch: thank you for the enlightening conversations on probability. To Yu-Hsuan Chen and Dennis Akos: thank you for helping with software-defined radios. And to the other GPS Lab members: thank you for the many interesting conversations and the presentation feedback.

To the anonymous reviewers who read my research publications submissions, thank you for helping me take my work to the next level.

To the navigation community, thank you for the warm welcomes, interesting conversations, and fun nights out.

To Christopher Mayhew and my other SpaceX colleagues, thank you for instilling in me a drive to test quickly.

To Charlotte Engstrom: thank you for hiring me as a legal intern that first summer, believing in me, and broadening my professional horizons.

To Debbie Poss: thanks for going above and beyond and inspiring me to pursue math.

To my advisors: Professor Todd Walter and Dr. Sherman Lo, when people ask me about doing a PhD, I tell them that a PhD is 90% your advisor: a great advisor makes a great PhD. You all are the best advisors one could have. I brag about you both all the time. Thank you for supporting me with the right balance of technical advice, direction, and the freedom to explore. Thank you for defending my work among our colleagues in our field.

To my mother: thanks for being a real-life hero, letting me ride out the pandemic at your place, and everything else. To my father: thank you for your role in shaping who I am today. To my brother, Jeff: thanks for being a person I genuinely enjoy spending time with.

To Becket: thank you for being the world's most supportive partner.

# Contents

# List of Algorithms

# List of Tables

# List of Figures

# Key Terms

**Chimera** A proposal to incorporate NMA and watermark ranging authentication into GPS's L1C signal [1, 2]. Chimera is short for Chips-Message Robust Authentication. 34, 36, 37, 38, 128, 129, 130, 131, 172, 175, 176, 177, 180, 181

**Combinatorial Watermarking Function** A function that uses an input cryptographic seed to enact a Combinatorial Watermark on a Global Navigation Satellite System (GNSS) ranging code. 43, 172, 174, 177, 180, 247

**Combinatorial Watermark** A GNSS ranging code watermark based on pseudorandomly inverting a small fixed number of chips within each ranging code (or sections or groups of ranging codes). xx, 2, 3, 43, 177, 178, 179, 214, 247, 248, 250

**cryptographic primitive** A low-level function underlying or used within a security protocol that provides security guarantees regarding computational work. 2, 4, 5, 9, 10, 11, 14, 16, 19, 30, 35, 41, 60, 61, 117, 214

**cryptoperiod** The length of time for which the cryptographic information applies or is utilized for security purposes. 153, 157, 158, 167

**Galileo** The GNSS constellation created by the European Union. 18, 34, 35, 36, 39, 40, 114, 119, 133, 134, 151, 152, 162, 243

**hash path** A collection of hash points related via repeated application of a cryptographic hash function for the purpose of authentication with TESLA. xxi, 20, 21, 22, 23, 24, 25, 26, 30, 32, 36, 40, 54, 56, 102, 107, 108, 109, 110, 111, 113, 114, 116, 120, 121, 122, 124, 125, 129, 130, 131, 132, 133, 134, 135, 138, 140, 144, 146, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 160, 161, 163, 164, 165, 167, 231, 246

**hash point** A *b*-bit integer that is usually the output of a cryptographic hash that is used to generate cryptographic information for the purpose of authentication with TESLA. xx, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 33, 34, 36, 40, 48, 50, 51, 52, 53, 54, 55, 56, 60, 61, 63, 65, 66, 69, 71, 72, 73, 74, 75, 78, 79, 81, 94, 102, 103, 105, 106, 107, 108, 109, 110, 111, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 132, 133, 134, 136, 139, 140, 141, 142, 143, 144, 145, 146, 148, 149, 150, 151, 153, 156, 160, 161, 162, 163, 170, 171, 174, 175, 178, 188, 196, 215, 218, 223, 231, 232, 243, 244, 246, 247, 248

**message authenticity** The security guarantee that the information was sent from the sender. 24, 29, 50, 60, 71

**message integrity** The security guarantee that the information was not manipulated in transit. 16, 24, 50, 60, 61, 69, 71, 73, 143, 150

**nonce** A random number (sometimes cryptographic random) that is used only once. 57, 70, 97, 99, 159, 160

**Rainbow Table** The relevant data structure in a Rainbow Table attack on a one-way function [77]. The adversary attempts to memorize all potential hash paths generated from a one-way function hoping that it will have the authentic hash path in memory before its use. This attack is defeated by incorporating salt into the one-way function. xxii, 14, 150, 154

**ranging code** A pseudorandom sequence broadcast by GNSS to allow receivers to deduce their range to a satellite. xx, 3, 4, 5, 13, 14, 17, 37, 38, 39, 40, 43, 53, 54, 55, 56, 111, 113, 114, 115, 116, 117, 118, 119, 120, 129, 135, 136, 137, 138, 169, 170, 172, 173, 174, 175, 176, 177, 178, 179, 180, 184, 187, 188, 189, 190, 196, 197, 203, 205, 209, 211, 213, 214, 220, 228, 230, 231, 232, 239, 241, 243, 244, 246, 248

**receipt safety** The security guarantee that a message arrived early enough within the TESLA framework. 24, 29, 41, 42, 60, 61, 62, 69, 71, 72, 73, 74, 75, 78, 81, 83, 84, 85, 91, 93

**salt** A salt is a random number used within a one-way function to defend against Rainbow Table Attacks. xxi, 14, 20, 23, 42, 149, 150, 151, 154, 155, 159, 160, 162, 248

**watermark** A cryptographic pseudorandom perturbation used to embed authentication security with limited perceptibility without privileged information. xx, 37, 115, 169, 246, 247, 248

# List of Acronyms

**ACAS** Assisted Commercial Authentication Service 40

**AES** Advanced Encryption Standard xxviii, 13, 15, 19, 38, 39, 176, 177, 180, 181

**AES-GCM** AES in Galois Counter Mode 19, 26

**AIRAC** Aeronautical Information Regulation and Control 68, 150

**AWGN** additive white Gaussian noise 189, 190, 195, 197, 203, 208, 213, 220, 241

**bps** bits per second 34, 35, 36

**BPSK** binary phase shift key 173, 211, 212, 213, 220, 225, 226, 228

**CA** certificate authority 153, 157, 164, 165

**CLT** Central Limit Theorem 180, 190, 195, 199, 200, 203, 206, 207, 210, 217, 218, 221, 222, 233, 237, 238, 241, 242

**CRC** cyclic redundancy check 140, 142

**DS** asymmetric digital signature 16, 23, 24, 25, 26, 31, 32, 33, 55, 56, 102, 103, 109, 110, 111, 112, 114, 116, 117, 118, 119, 120, 121, 122, 124, 125, 129, 130, 133, 135, 138, 148, 149, 151, 152, 153, 154, 156, 157, 159, 160, 161, 162, 164, 166

**ECDSA** Elliptic Curve Digital Signature Algorithm 17, 19, 34, 37, 38, 55, 56, 103, 104, 128, 130, 154, 157, 158, 159, 160, 164, 165, 166, 176, 248

**EGNOS** European Geostationary Navigation Overlay Service 133, 167, 230

**GIC** onboard GNSS-independent clock xxix, 28, 29, 32, 33, 41, 42, 47, 48, 49, 50, 51, 52, 54, 55, 57, 58, 59, 60, 61, 62, 63, 64, 65, 67, 68, 69, 70, 71, 72, 73, 74, 76, 77, 78, 79, 80, 83, 85, 86, 92, 94, 96, 100, 101, 131, 132, 136, 196, 209, 245, 246, 247, 248

**GNSS** Global Navigation Satellite System iv, v, xx, xxi, 1, 2, 3, 5, 6, 7, 8, 11, 12, 13, 14, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 47, 48, 49, 50, 53, 54, 55, 56, 59, 60, 61, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 77, 83, 92, 96, 97, 98, 102, 103, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 122, 124, 125, 127, 128, 129, 130, 131, 132, 133, 135, 137, 138, 139, 140, 143, 146, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 160, 161, 162, 169, 171, 172, 174, 175, 177, 181, 188, 190, 192, 197, 198, 200, 209, 210, 218, 220, 231, 233, 234, 237, 239, 242, 244, 245, 246, 247, 248, 249, 250

**GPS** Global Positioning System iv, xx, 34, 39, 114, 115, 116, 128, 135, 172, 175, 187, 188, 189, 193, 196, 203, 204, 205, 220

**HDSCER** hard-decision security code estimation and replay 214, 215, 216, 217, 218, 220, 221, 222, 223, 225, 226, 227, 228, 229, 239, 240, 249

**HPE** hash path end 21, 23, 24, 25, 26, 32, 104, 111, 120, 148, 149, 150, 151, 152, 153, 154, 158, 160, 161, 164, 165, 166

**I** in-phase 35

**ICAO** International Civil Aviation Organization 68, 150

**ICD** interface control document 134, 153, 157, 162

**IOD** issue of data 146, 147

**IV** initialization vector 13

**LEO** low-earth orbit 1, 2, 8, 31, 41, 120, 137

**LTI** linear time-invariant 187, 188, 190, 191, 196, 198

**MAAST** Matlab Algorithm Availability Simulation Tool 167, 168, 248

**MAC** message authentication code xxviii, 16, 17, 18, 26, 51, 65, 102, 139, 140, 141, 147, 174

**MT** message type 36, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 163, 164, 165, 166, 167, 168, 232

**NIST** National Institute of Standards and Technology 15, 16, 30, 154

**NMA** navigation message authentication xx, 35, 36, 37, 38, 39, 139, 146, 174, 176, 178, 232, 234, 237, 247

**NTP** Network Time Protocol 56, 59, 92, 95, 97, 98, 99

**NTS** Network Time Security 56, 57, 58, 59, 69, 70, 83, 86, 87, 89, 92, 93, 95, 96, 99, 100, 101, 246

**OSNMA** Open Service Navigation Message Authentication 34, 35, 36, 40, 104, 119, 134, 162, 243

**OTAR** over-the-air rekeying 42, 153, 154, 157, 158

**PDF** probability density function 212

**PFA** probability of false alarm 187, 200, 206, 207, 208, 222, 224, 231, 233, 234, 235, 238, 239, 240, 242

**PKI** public key infrastructure 30, 153, 154

**PMD** probability of missed detection 176, 180, 187, 200, 205, 206, 207, 208, 222, 223, 224, 231, 233, 234, 235, 236, 237, 238, 239, 242

**PNT** Positioning, Navigation, and Timing 2, 3, 5, 6, 7, 8, 29, 32, 34, 38, 47, 48, 53, 54, 61, 113, 115, 120, 127, 128, 171

**PRS** Public Regulated Service 114

**PSD** power spectral density 179

**Q** quadrature 35

**QPSK** quadrature phase shift key 193, 194

**RAIM** receiver autonomous integrity monitoring 5

**RECS** Re-Encrypted Code Sequence 40

**RoT** Root of Trust 157

**SBAS** Satellite-based Augmentation System 18, 34, 35, 36, 42, 68, 103, 107, 128, 132, 133, 137, 138, 139, 140, 143, 144, 146, 147, 148, 163, 164, 165, 166, 167, 172, 230, 231, 232, 234, 238, 239, 240, 241, 242, 243, 247, 248, 250

**SCER** Security Code Estimation and Replay 6, 7, 38, 43, 44, 172, 173, 177, 179, 196, 197, 199, 200, 204, 205, 206, 207, 209, 210, 211, 214, 221, 226, 227, 230, 233, 234, 235, 239, 240, 245, 248, 249, 250

**SDR** software-defined radio 203, 204, 205, 240, 241, 243

**SNR** signal-to-noise ratio 192, 207, 212, 213, 221, 222, 223, 225, 226, 227, 240, 248, 249

**TESLA** Timed Efficient Stream Loss-tolerant Authentication iv, xx, xxi, xxii, xxviii, xxix, 3, 11, 20, 21, 22, 23, 24, 26, 27, 28, 29, 30, 31, 32, 34, 35, 36, 37, 38, 40, 41, 42, 43, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 59, 61, 63, 67, 68, 69, 71, 72, 77, 78, 79, 80, 92, 102, 103, 105, 111, 113, 114, 118, 119, 121, 124, 128, 129, 130, 132, 133, 138, 139, 140, 141, 143, 146, 148, 149, 150, 151, 153, 155, 156, 157, 159, 161, 162, 163, 164, 165, 166, 167, 169, 174, 175, 177, 178, 186, 231, 232, 242, 243, 244, 245, 246, 247, 248

**TFAF** time to first authenticated fix 31, 32, 43, 105, 150, 155, 156, 157, 158, 159, 162, 163, 164, 166, 167, 168, 246, 250

**TTA** time to authentication 2, 31, 32, 33, 34, 35, 39, 41, 42, 47, 55, 63, 65, 68, 77, 79, 104, 105, 108, 109, 110, 111, 112, 116, 117, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 136, 137, 138, 141, 143, 148, 150, 175, 209, 231, 232, 234, 245, 246, 247, 248, 249

**WAAS** Wide Area Augmentation System 162, 167, 193, 230, 231, 240, 241

# List of Symbols

AES-CTR Advanced Encryption Standard (AES) in counter mode 13

CMF commitment-MAC function 16, 22, 26, 66, 139, 174, 178

CMT commitment-MAC tag 16, 22, 23, 24, 27, 28, 29, 32, 36, 50, 51, 52, 65, 66, 67, 71, 72, 78, 79, 80, 102, 103, 105, 106, 107, 108, 109, 110, 112, 113, 118, 129, 130, 161, 246, 247

DEC block cipher decryptor of block cipher (ENC) 12

ENC block cipher xxviii, 12, 13, 19, 118

ENC-CTR block cipher in counter mode 13

ENC-GCM block cipher in Galois counter mode 13

H cryptographic hash function 15, 19, 20, 144

HMAC keyed-hash message authentication code 16, 17, 18, 19, 35, 36, 104, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 162, 163, 182, 184, 186, 232

$p$ hash point 20, 21, 106, 144, 175

HPF hash path function 20, 21, 22, 23, 24, 56, 111, 116, 117, 120, 139, 144, 150, 151, 160, 161, 162

$I$ the interval of time for which a TESLA hash point applies 32, 33, 123

KDF key-derivation function 12, 13, 16, 19, 22, 37, 105, 106, 107, 111, 114, 115, 116, 117, 118, 119, 120, 124, 129, 135, 139, 142, 146, 162, 169, 172, 174, 175, 178, 184, 186, 231, 232

$L_{\mathbf{hp}}$ the length of time needed to transmit a hash point 32, 33, 34, 125, 130

NM navigation message 18, 146, 147

PRF pseudorandom function 4, 5, 11, 12, 13, 54, 55, 56, 113, 114, 115, 116, 117, 135, 136, 138, 169, 170, 176

PRN pseudorandom noise code assignment 107, 142, 147, 174, 175, 176, 203, 204, 220

$\theta$ the clock offset of the GIC 50, 68, 73

$\Theta$ the TESLA commitment reveal delay 32, 33, 50, 52, 59, 60, 61, 63, 64, 65, 67, 68, 72, 73, 77, 78, 130, 232

# Chapter 1

# Introduction

> The time to repair the roof is when the sun is shining.
>
> John F. Kennedy

As satellite launch costs rapidly decrease for low-earth orbit (LEO), new opportunities arise to leverage LEO for satellite navigation [85]. At the same time, vulnerabilities remain with the security of current Global Navigation Satellite System (GNSS) signals we all use daily, and challenges arise with incorporating authentication security mechanisms [91]. With the right expertise, anyone could broadcast false GNSS signals causing disruption, harm, or loss of life in transportation technology accustomed to safety and reliability. A new generation of GNSS presents a new opportunity and treatment for the cryptographic authentication of GNSS.

GNSS security vulnerabilities result from anyone being able to broadcast a false GNSS signal to fool a receiver. Many spoofing mitigation techniques have been explored [44, 83]. For instance, one could utilize the signal direction of arrival to mitigate some spoofing threats [86]. Suppose a receiver notices that all of the GNSS are coming from a single direction on the ground rather than from the correct overhead geometry. Then, the receiver has strong reason to believe that the signals are not authentic. While this idea provides excellent mitigation for the spoofing threat most accessible to malicious actors, it does not cover more advanced adversaries, such as those broadcasting from multiple locations or outer space.

This thesis focuses on cryptographic authentication strategies to mitigate spoofing. Cryptographic authentication security leverages arguments about the difficulty of solving mathematical problems and the adversary's computational resources. To break the security of cryptographic primitive functions, the adversary needs more computers and time than what is physically achievable. Yet, cryptographic authentication is still no silver bullet for GNSS, as it has vulnerabilities inherent to how Positioning, Navigation, and Timing (PNT) works. Because the arrival time of the GNSS determines the PNT measurement, and cryptography does not provide security against delays [16], cryptographic authentication is still a mitigation strategy rather than a provable-secure strategy. But, incorporating cryptography into GNSS makes spoofing attacks extremely difficult and offloads provable security to arguments about the receiver's time synchronization.

For the LEO GNSS designer, this work provides a comprehensive guide on how to design cryptographic systems for GNSS, covering efficient cryptographic constructions, necessary time synchronization protocols, and signal watermarking, among other things. PNT with authentication burdens the signal data bandwidth. By leveraging the design principles of this work, the current GNSS designer can shrink the required data bandwidth needed for authentication. PNT with authentication will introduce a delay in receivers utilizing PNT measurements. As new technology alleviates the data bandwidth concern in the future, the most significant GNSS authentication consideration will be time to authentication (TTA). By leveraging the design principles of this work, the future GNSS designer can shrink the TTA.

For the GNSS authentication researcher, this work provides a new treatment for GNSS authentication, covering how to show cryptographic construction correctness with complex authentication structures, proving necessary time synchronization protocols, and proving the security of signal watermarking, among other things. By leveraging the mathematical constructions of the Combinatorial Watermark, there are pathways for more rigorous notions of security of receiver processing statistics. The techniques herein can aid with the security research of present-ranging authentication schemes.

The rest of the introduction covers GNSS threat models, relevant modern Cryptography, and GNSS Authentication. Thereafter, I organize the following into individual chapters. The first chapter covers time synchronization for GNSS Timed Efficient Stream Loss-tolerant Authentication (TESLA). Its importance and my experience informed my intentional decision to make this chapter first; do not skip over it. The next chapter covers efficient TESLA constructions by utilizing a temporal-geometric interpretation of the underlying cryptographic objects. The chapter after that covers efficient maintenance data in support of GNSS TESLA. Then, I finish with a complete treatment of the Combinatorial Watermark: the mathematical constructions and the analysis of watermark-induced receiver statistics.

But first, I will start with a quick treatment on the basics.

## 1.1 GNSS Range Processing

For GNSS to enable receivers to determine PNT, the GNSS must manage a constellation of satellites that produce a specific type of signal that enables the receiver to measure each signal's time of flight from satellite to receiver. Each time of flight provides a measurement of each satellite's range to the receiver for incorporation into the Trilateration Problem to find a PNT solution. This section briefly overviews how range signal processing works in the context of cryptographically-generated signals. For a proper treatment on how GNSS range signal processing works, I refer to [24].

Good-ranging signals are pseudorandom-sequence codes with very high autocorrelation and low cross-correlations. The ranging code is the pseudorandom sequence, and the signal will include the ranging code and optionally include additional data modulated with the ranging code and is usually modulated onto a carrier wave. Signals that derive from pseudorandom sequences generate signals with these two properties. Many of today's GNSS ranging codes use pseudorandom sequences that have other advantageous properties (e.g., bounded cross-correlations, computationally easy generation), and there are several GNSS that use pseudorandom sequences resulting from the encryption of the signals themselves.

To measure the range of a satellite, the receiver will compute a replica of the

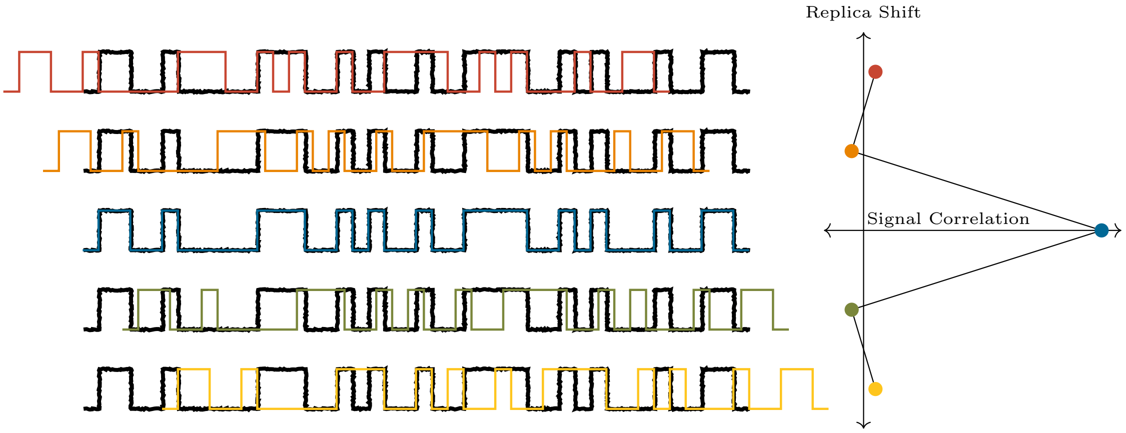


Figure 1.1: Conceptual Diagram Of GNSS Range Signal Processing.

In black is a noisy ranging signal composed of a pseudorandom sequence. A shifting replica is correlated with the noisy signal recorded by the receiver. In the diagram, each shift is its own color. Because the signals are pseudorandom, the replica will have a high correlation when the replica and signal are aligned; otherwise, there will be low correlation. The shift with the high correlation indicates the time of flight of the ranging signal.

pseudorandom ranging code. Then, the receiver will slide shift its replica across the noisy signal recorded by its radio. Fig. 1.1 provides a conceptual diagram of the process. As depicted in Fig. 1.1, when the shifted replica and the noisy signal are aligned, the correlation will be high, and otherwise, the correlation will be low. The shift with that high correlation measures the signal flight time from the satellite to the receiver. This measurement is called a pseudorange since it measures the time of flight with other effects (discussed in [24]) rather than the range directly.

When using a cryptographic signal, the pseudorandomness derives from a cryptographic pseudorandom function (PRF). These PRF signals result from the encryption of a signal or a cryptographic construction from Chapters 3 and 5. Because of the security guarantees of the cryptographic primitive underlying the PRF, the adversary's best prediction (without knowledge of a cryptographic secret) of the cross-correlation with any replica sequence and particular replica shift will be a Bernoulli distribution. Because the Bernoulli distribution has a quickly decaying upper tail, the adversary should not be able to guess a PRF-ranging code that can produce a high correlation with the replica on the receiver. If the adversary could generate a sequence with a high correlation with a PRF-generated ranging code, then that adversary could predict the output of the PRF, which is asserted to be impossible by the underlying cryptographic primitive. However, other threats against GNSS ranging processing are discussed in Section 1.2.

## 1.2 GNSS Threat Models

GNSS constellations generally have some assortment of two types of signals: data and ranging. The data signal transmits the data necessary (but not limited to) for a receiver to compute the satellite positions and corrections. The ranging signal enables the receiver to deduce its range to each satellite. I refer to [24] on using the data and ranges to compute a PNT measurement.

For the most part, GNSS signals are open. Their specifications and simulators are readily available online, meaning anyone can potentially generate false GNSS signals to manipulate a receiver's PNT measurement. The consequences of spoofing can include anything from annoying disruptions to deadly events in systems that assert safety-of-life requirements. This section briefly describes different adversary capabilities relevant to this work.

Fig. 1.2 provides a conceptual diagram of the adversaries relevant to this work. I refer to [83] for a more complete taxonomy of adversaries.

GNSS security is a game of cat and mouse, as provable-ranging security of PNT remains elusive [16, 58, 68]. To begin, a single spoofer with knowledge of the signal specification can generate a false signal and broadcast it to the receiver. From a cryptographic security point of view, since the receiver knows the signal specification to use the signal, the adversary also knows the signal specification. The receiver can counter with several strategies to detect a naive spoofer. For instance, the receiver can use PNT solution residuals (e.g., receiver autonomous integrity monitoring (RAIM)) and signal processing (e.g., power, Doppler frequency) checks to reject potentially spoofed signals.

The spoofer counters by ensuring that its spoofed signals are consistent (i.e.,

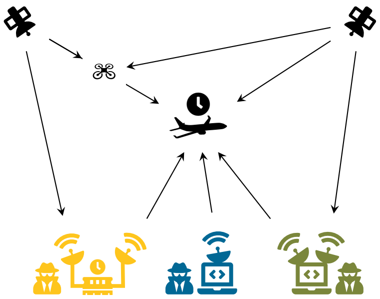


Figure 1.2: Conceptual Diagram Of Three Relevant Types Of Adversary Threat Models.

An adversary could engage in a naive spoof where no attempt is made to observe and replay the cryptography on the signal, but the spoof is a consistent signal (**blue**, bottom center). An adversary could engage in meaconing where the signal is directionally observed and replayed with delays with limited signal processing (**yellow**, bottom left). The diagram represents this with a simple memory stick that stores and replays the signal data without other manipulation. When the meaconing attack does not attempt to delay each satellite signal individually, then the spoofed position will be the position of the spoofer (e.g., quadcopter, top left). An adversary could engage in an SCER attack where the adversary uses signal processing to directly observe the cryptography of the signal and regenerate a signal (**green**, bottom right). For meaconing and SCER attacks, while the cryptography, noise conditions, and other mitigations (e.g., signal directional and PNT jump detection) would mostly be effective, the receiver time synchronization is the only pathway for provable deterrence with arguments regarding the speed-of-light lower bound transit delay.

the spoofed PNT estimate is mathematically consistent, and the spoofed signal has nothing that would disqualify it during signal processing). The receiver counters by upgrading its antenna to a directional antenna, enabling signal-directional checks. The spoofer counters again by using multiple (potentially flying) transmitting antennae to ensure the signal comes from plausible directions. This adversary is depicted in Fig. 1.2: **blue**, bottom center. While the work required of this spoofer is already complex, there is no proof that such an adversary is impossible.

This work focuses on the cryptography of the signal. In Section 1.3, I discuss how cryptographic systems assemble games that are proven impossible for an adversary by asserting that the adversary needs inordinate computational resources.

And so now, the GNSS provider and receiver together counter by augmenting the signal with cryptographic information and requiring cryptographic checks when receiving the signal. Section 1.4 describes several strategies for incorporating cryptography into GNSS, such as incorporating signatures and watermarks into the signal. But then, sadly, the spoofer counters by observing the cryptographic information in the signal and replaying the cryptographic signal. Because manipulating signal arrival time manipulates the PNT solution, the adversary can manipulate the deduced PNT solution without knowledge of the cryptographic secrets underlying the cryptographic authentication security arguments. These adversaries include the two adversaries in Fig. 1.2 with ***receiving and transmitting*** radios. While the work required of this spoofer is even more complex, requiring technical equipment and know-how, from a cryptographic security point of view, there is no proof that such an adversary is impossible. At issue is that authentication cryptography protects against data manipulation but not delays [16].

In a meaconing attack, the adversary listens and repeats the signal with little or no signal processing beyond a delay (Fig. 1.2: yellow, bottom left). The signal's cryptographic protections are simply repeated to the receiver ***without any knowledge*** of the existence of the cryptographic protections. By artificially increasing the pseudoranges of individual satellites, the spoofer could induce a near-arbitrary PNT solution on the receiver. In a Security Code Estimation and Replay (SCER) attack (Fig. 1.2: **green**, bottom right), the adversary directly attempts to observe the cryptographic protections and then replays the observed cryptographic protections. The receiver counters these adversaries by implementing signal arrival checks to catch delayed signals. The adversary counters by reducing its latency to 0, bumping against the fundamental limitation of the speed of light (noting that the speed of light is slower when traversing the atmosphere).

As a thought experiment, a potential ultimate adversary could be modeled as the following. The adversary is a rich, GNSS expert under house arrest with a GNSS anchor bracelet. This adversary wins the security game if they can breach the conditions of their house arrest while still reporting a compliant PNT solution. The adversary can craft an ankle-worn Faraday cage with internal directional antennas. The adversary can access a massive listening antennae apparatus to observe and replay the cryptographic protections. The replay apparatus is expensive enough to ensure no latency anywhere in the system. While this rich, willing, and able adversary is ridiculous, and I expect authorities would notice this adversary wearing a large apparatus on their ankle, from a cryptographic security point of view, there is no proof that such an adversary is impossible.

With every protection mentioned so far, there is a more sophisticated adversary. However, theoretically, if the honest receiver perfectly knew its PNT, then it could perfectly detect delayed signals against provable conditions based on the speed of light limitation, but that poses a Catch-22 on the problem GNSS Authentication wishes to solve. As more imperfect checks are added to the receiver and the checks become more stringent, the probability of a false alarm will increase. Moreover, receivers will venture into non-ideal signal conditions frequently. We run into the danger of making receivers unnecessarily brittle to a rare threat. So should we give up?

No! The system must balance the threat and mitigations. I am concerned primarily with two situations. First, an adversary spoofs the PNT solution of an airplane on approach while landing, inducing an instrument approach to crash. Second, an adversary spoofs an autonomous vehicle into colliding with another vehicle or object on a roadway.

To counteract these dangerous vulnerability scenarios, I would recommend the following mitigations. First, receivers should have a directional antenna [86]. Second, roadways and airports should implement monitoring of GNSS signals [29, 61]. Third, the signal should employ data and ranging cryptographic authentication.

While not provably safe, I expect the required multi-directional radio equipment will be observable by the authorities. And if not, the space of attainable arbitrary PNT spoofs capable of causing a vehicle to crash would be minimal. While I support all of these mitigations and other mitigations, this work focuses on prudent cryptographic design choices future GNSS, public and private LEO, constellations should take.

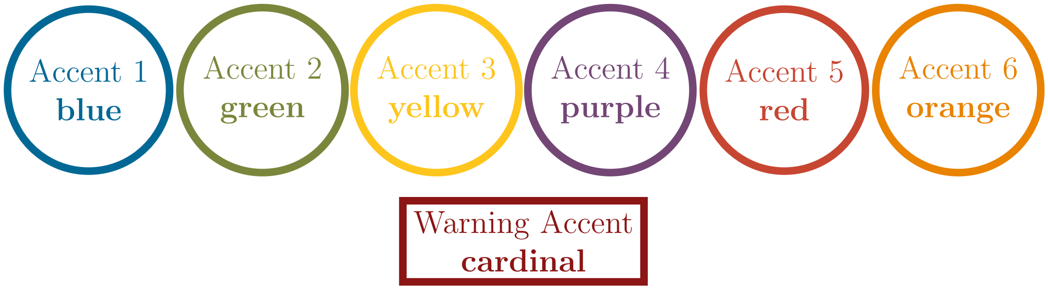


Figure 1.3: Accent Colors Utilized In This Thesis.

### 1.2.1 Accent Colors

This thesis utilizes accent colors extensively. These colors primarily show the correspondence of objects within conceptual diagrams. The colors are set programmatically, meaning that color-accessible versions (for those with color blindness) are available. By default, the colors are from the Stanford University Color Identity Guide in a color configuration attempting to be accessible to those with the most common types of color blindness.

Fig. 1.3 provides the colors utilized in this thesis in order of utilization: Accents 1 and 2 are utilized more often than Accents 5 and 6. The colors available are designed so that adjacent colors are easily distinguishable for those with the most common types of color blindness (e.g., 1-2-3, 2-3-4, 5-6-1). Moreover, the warning accent color is easily distinguishable from Accents 1 and 2. However, the accent colors can be changed programmatically for further color accessibility.

## 1.3 Modern Cryptography

Modern cryptography employs cryptographic primitives to assert security conditions on information. In this work, the most relevant security condition relates to the authenticity and integrity of information. Information is authentic if it is known to come from a specific source. Information has integrity if it is known not to have been manipulated in transit (i.e., not manipulated in the delayed sense, see Section 1.2).

The underlying cryptographic primitives are designed around assumptions of the computational hardness of solving mathematical problems without privileged information. For instance, the privileged information could be a private cryptographic key. The mathematical problem could be to generate a valid cryptographic signature. The receiver knows information with a valid signature is authentic and integral because forging a cryptographic signature requires inordinate computational resources.

The assumptions underlying computational hardness relate to the existence of an efficient algorithm to solve a mathematical problem. By relying on the hardness of certain mathematical problems, modern cryptography need not rely on the obscurity of any underlying mathematics. In cryptographic terminology, a secure cryptographic algorithm will ***admit*** no efficient algorithm to solve a mathematical problem without the protocol's privileged information. The adversary will attempt to break the security of a cryptographic algorithm by solving the mathematical problem or ***exhaustively*** guessing the privileged information. Typically, when the cryptographic primitive admits no efficient algorithm, the adversary's best algorithm is to exhaustively guess a solution to the mathematical problem. Cryptographic security relies on (1) assuming a best, known algorithm (usually, and for this work, exhaustive search) and (2) increasing the search space of the privileged information until an exhaustive search is infeasible with any reasonable computer. A typical cryptographic proof of security utilizes proof by contradiction, where breaking the security of the system requires breaking the security of the underlying cryptographic primitive. A cryptographic function is broken if someone discovers an efficient algorithm capable of decreasing the effective security level significantly enough to enable a reasonable computer to solve the mathematical problem without the privileged information.

The cryptographic security level of a protocol measures the expected number of computer operations required to solve the underlying mathematical problem enabling a security condition. The cryptographic community presently considers a 128-bit security level enough for modern computers. 128-bit security means that an adversary should expect to complete $2^{128}$ operations before breaking the security afforded by a cryptographic primitive. $2^{128}$ operations is enormous. For intuition, at the time of writing, the highest observed global Bitcoin hash rate is on the order of $8 \cdot 10^{20} \approx 2^{69}$ hashes per second. This means it would take the world's computers working together, on expectation, $2^{59}$ seconds (18 trillion years and beyond the expected heat death of the universe) to solve a cryptographic hashing math problem at the 128-bit security level.

Historically, people use cryptography to ensure the ***secrecy*** of information in transit rather than its ***authenticity*** and ***integrity***. People broadly understand that encryption with a cipher provides secrecy. However, not as widely understood, secrecy and authenticity/integrity (or encryption and authentication) are orthogonal: authentication does not provide secrecy, and ***encryption does not provide authenticity nor integrity***. In the context of this work, encrypting a GNSS signal does ***not*** provide any authentication security. I refer to Section 1.4.4 for further discussion.

The following sections discuss the relevant cryptography for GNSS. In Section 1.3.1, I discuss the background of the cryptographic primitives relevant to this work, and Section 1.3.2 discusses security levels in the GNSS context. I list suggested cryptographic primitives in Section 1.3.3. Section 1.3.4 discusses the background of the primary authentication protocol of this work. In this work, I utilize my own terminology to describe TESLA, and Section 1.3.5 defends my choice. Section 1.3.6 discusses the background of an important time synchronization requirement for ***any*** GNSS ranging authentication protocol. The descriptions below provide the appropriate rigor for this work's introduction, based on [28]. I refer to [28] for those seeking more information on Cryptography.

### 1.3.1 Cryptographic Primitives

From the following cryptographic primitives, one can construct protocols that provide the security conditions of later subsections.

A pseudorandom function (PRF) is a deterministic dual-input-single-output function: for instance, $y = \text{PRF}(k, x)$ whose output appears random. The first input is usually called a key. Suppose that key $k$ is randomly chosen from a large key space and held secret. For a secure PRF, the adversary has a negligible advantage (over a random guess) in predicting the output of PRF for any set of adversarial-chosen inputs $x$. Because an adversary could attempt to memorize the inputs and outputs of PRF, the key space size must be large enough to deter an attempt. For the subject of this work, one can practically construct a PRF from a modern block cipher or a cryptographic hash function, noting that there are other ways to construct a PRF.

A key-derivation function (KDF) enables one to derive additional pseudorandom keys from another key, sometimes called key lengthening or a sub-key derivation function in cryptography contexts (not to be confused with problems related to deriving a cryptographic key from a human-generated password). One can also construct a KDF from a modern block cipher or cryptographic hash function. In this work, a KDF will typically be used to derive many subkeys, each assigned a context, from a main key. For instance, $k_{\text{context}} = \text{KDF}(k_{\text{main}}, \text{context})$. KDF enables efficient use of Global Navigation Satellite System (GNSS) data bandwidth by transmitting only the main key for receivers to derive all the other context keys. Moreover, using KDF-derived subkeys (rather than using a main key in multiple places) prohibits related-key attacks by ensuring the subkeys are cryptographically independent [20, 25, 81].

While PRF and KDF can be constructed with either a block cipher or a cryptographic hash function, one should consider two design considerations. First, block ciphers and cryptographic hash functions were designed to meet specific use cases. For instance, hash functions were designed to digest large inputs, whereas block ciphers fixed block sizes. Second, the receiver hardware may have hardware accelerators for specific functions, or the functions may require different amounts of power to operate. The GNSS designer must balance the hardware with the necessary mathematical construction to achieve its design goals.

### Encryption with Block Ciphers

A block cipher (ENC) is a deterministic dual-input-single-output bijective function that appears random. The first argument is usually called a key, and given the key, $\text{ENC}(k, \cdot)$ is a permutation function. Accompanying the cipher is its block cipher decryptor of ENC (DEC) function, which will act as the inverse permutation function with the key. For this work, I explicitly specify block ciphers because they are state-of-the-art at the time of writing, fulfilling several security properties beyond non-block ciphers. Moreover, hardware accelerators for the block cipher Advanced Encryption Standard (AES) are ubiquitous. Among the properties of a secure block cipher is that the output of $\text{ENC}(k, \cdot)$ should be indistinguishable from a random function, making ENC a PRF. Of particular note is that ENC, as typically used, is not a one-way function, a pertinent property of cryptographic hash functions. Block ciphers have several different modes of operation. This work will use block ciphers as a KDF and a traditional cipher in Counter Mode and Galois Counter Mode.

A block cipher in counter mode (ENC-CTR) encrypts an incrementing counter and then xors that output with the plaintext to produce the ciphertext. The counter starts at an initialization vector (IV). A notable advantage of ENC-CTR, practically AES in counter mode (AES-CTR), is its speed in hardware. A block cipher in Galois counter mode (ENC-GCM) encrypts and authenticates the information, which is helpful to someone who wants to ensure the secrecy and authenticity of a message with one off-the-shelf function.

### Hiding Commitments

For a public ranging system to function, the receiver must have prior knowledge of the pseudorandom sequence ranging code. The GNSS provider, in essence, ***commits*** to utilizing a specific ranging code for the receiver to complete the shifted-replica correlation from Section 1.1.

In the cryptographic sense, a bit commitment is a procedure that involves two phases: the commit phase and the reveal phase. The protocol involves two pieces of information: the commitment and the proof. In the commitment phase, the GNSS provider commits to a ranging code without revealing what that ranging code is. In the reveal phase, the GNSS provider reveals ***hidden*** information and a proof that proves that the provider utilized that revealed information to generate the commitment. A cryptographic commitment is binding, meaning that when GNSS commits to a ranging code with the hidden information, GNSS cannot produce a proof that works with the commitment based on ***other*** hidden information. The GNSS could not have generated two ranging codes that satisfy the aforementioned proof as GNSS committed to a particular ranging code. The binding commitment property is not very useful to the authentication context because I reasonably assume that the authentic and honest GNSS provider will not want to break its commitments.

However, GNSS requires using ***hiding*** commitments. The hiding commitment property ensures that the commitment reveals nothing about the hidden information from which the commitment was derived. GNSS authentication requires hiding commitments because the adversary could attempt to memorize every input and output of the commitment function primitive with a Rainbow Table Attack [77]. The adversary wins when they assemble a large enough Rainbow Table that it is likely that GNSS will utilize a commitment that is already in its Rainbow Table. To deter this attack on the commitment function (and thereby make it a hiding commitment function), the commitment function must incorporate a random salt. One can construct the authentication protocol in Section 1.3.4 with hiding bit commitments.

One can create a hiding commitment function from a ***salted*** cryptographic hash function. A cryptographic salt is a cryptographically random bit-string, selected independently of any other secrets of accompanying cryptographic protocols, incorporated into the cryptographic primitive. Suppose GNSS incorporates a cryptographically random $b$-bit salt (i.e., literally a random integer) into the hash function. Essentially, GNSS will draw a random hash function among a hash function family of size $2^b$. Then, the rainbow-table adversary must assemble $2^b$ Rainbow Tables to anticipate the salt. Once the salt is known, the adversarial game returns to generating a single Rainbow Table. However, provided the salt is released not too far before its use, the salt is used over a short interval, and the salt integrity is assured with authentication, the salt provides a hiding commitment and mitigates the Rainbow Table Attack. A 128-bit salt is sufficient for any application; however, one can decrease the amount after careful probabilistic analysis [32].

One also needs a preimage-resistant hash function to construct an authentication protocol based on hiding commitments.

### Preimage-Resistant Cryptographic Hash Functions

A cryptographic hash function (H) is a deterministic function that maps a large message space to a small digest space with properties such as collision resistance and preimage resistance. A hash function is collision-resistant if any adversary has a negligible probability of ever producing an $m_0$ and $m_1$ such that $\mathrm{H}(m_0) = \mathrm{H}(m_1)$. Recently, Google demonstrated that it could break the collision-resistant property of the hash function SHA-1 based on prior theoretical attacks, necessitating deprecating SHA-1 [106]. Since then, SHA2-256 has become the de facto standard cryptographic hash function, and National Institute of Standards and Technology (NIST) proactively standardized SHA3 [103, 105].

Suppose the adversary is provided $h = \mathrm{H}(m_0)$ where $m_0$ is random and not provided to the adversary. A hash function is preimage-resistant if any adversary has a negligible probability of ever producing another $m_1$ so that $\mathrm{H}(m_0) = \mathrm{H}(m_1)$. Whereas collision-resistance is when an adversary can produce two messages with the same hash, preimage-resistance is when an adversary can invert the hash function. A preimage-resistance function is sometimes called a one-way function.

Collision resistance suffers from Birthday Attacks, but preimage resistance does not. Therefore, with a secure cryptographic hash function that produces a 256-bit digest, the collision-resistance security level is 128-bit. For the same function, the preimage-resistance would be 256-bit. In this thesis, there will be times when only the preimage resistance of H is required, and so the output will be truncated to 128 bits.

The cryptographic hash functions endorsed by NIST, such as SHA2-256, are ubiquitous and standard. Because those function digests exceed 128 bits and digests can be left-truncated [97], I will use the NIST-hash functions left-truncated to amounts depending on the context. In those cases, the security level will decrease by the amount of truncation (e.g., for SHA2-256, 256-bits to 128-bits) for modern computers.

For those resistant to utilizing SHA for hashing, particularly for Section 1.3.4 where large amounts of repeated, sequential hashing are required, it might be advantageous to construct a cryptographic hash function from AES (or another block cipher). The most well-studied AES-based hash function is ECHO [21]. While ECHO is not known to be broken, it did not pass to the finalist round in the NIST SHA3 competition. Great care should be taken when electing not to utilize standard cryptographic functions well studied by the cryptography community.

### Symmetric Authentication with CMF and CMTs

For this work, the message authentication code (MAC) cryptographic primitive must also produce a commitment. At the time of writing, keyed-hash message authentication code (HMAC) is the de facto standard for message authentication codes, which already satisfies the additional commitment-producing requirement. In the generalized context, I will use a MAC function that provides a commitment a commitment-MAC function (CMF). CMF produces a commitment-MAC tag (CMT).

CMF is a dual-input-single-output function that constructs a message authentication code from an underlying hash function (e.g., HMAC with SHA2 function). The first input is usually called the key, and the second is the message. If the sender and receiver securely distribute a key, CMF can ensure message integrity. Moreover, the digest provided by HMAC can be left-truncated to provide security according to the digest length. CMF provides a secure KDF. In this work, I will frequently use CMF for two purposes: to generate CMTs on messages for message integrity and to generate additional keys as a KDF.

### Asymmetric Authentication with Digital Signatures

With asymmetric authentication, there is a private signing and a public verification key. The sender uses its private signing key to generate an asymmetric digital signature (DS), and the receiver uses the sender's public key to verify the message with the DS. An asymmetric authentication protocol that utilizes DSs requires three algorithms. First, an algorithm must generate a public key from a private key. The key-generation algorithm must be one-way so that an adversary cannot learn the private key from the public key. Second, an algorithm must generate a signature from the private key and message. Third, a deterministic algorithm must always accept a public key and messages with signatures generated with the private key and reject otherwise.

At the time of writing, the Elliptic Curve Digital Signature Algorithm (ECDSA) is the de facto standard. Unfortunately, to provide 128-bit authentication security, ECDSA signatures require substantial bits to distribute public keys and digital signatures. For ECDSA to provide 128-bit authentication security, the compressed public key will require about twice the security level of bits, and the digital signature will require about four times the security level of bits. Another protocol called EC-Schnorr provides digital signatures that are slightly smaller than ECDSA signatures, but the protocol has not been standardized [90]. However, Bitcoin has adopted the protocol. Because of the data bandwidth limitation of GNSS, it is not feasible to sign every GNSS message and ranging code with ECDSA (or EC-Schnorr). This issue motivates utilizing HMAC with the protocol of Section 1.3.4 and the entirety of this dissertation.

### 1.3.2 Cryptographic Security Levels

Cryptographic security arguments rely on the notion of inordinate computational resources. The reason why something is secure is because the required computation exceeds that achievable by any reasonable adversary. The cryptographic security level describes that notion of computational resource. The standard in modern cryptographic security is 128-bits. 128-bit cryptographic security means that an adversary should need $2^{128}$ of computational resource (e.g., number of operations, amount of memory) to break the security claim. $2^{128}$ is so large that even a galactic civilization should not be capable of performing computation at that level with current computational resources.

#### MAC Truncation

Because (1) GNSS is data bandwidth constrained and (2) GNSS users may require less security than 128-bits, a common accommodation is to left-truncate MACs [48]. Left truncation is permitted per [98], provided careful risk analysis is performed and that the GNSS specifies a maximum number of failed MAC verifications.

GNSS generally have very rigid message schedules. For instance, the Satellite-based Augmentation System (SBAS) transmits exactly one message per second. The security designer needs not consider a scenario where the adversary may send numerous perturbed copies of a navigation message (NM) in an attempt to submit a single forgery. This means that GNSS should prescribe that each authenticated information has exactly one attempt to pass verification.

Conveniently, in this GNSS context, this means that a $b$-bit MAC has $b$ security. In other words, the probability that an adversary can generate a forgery is $2^{-b}$, or the expected number of adversary attempts before the adversary generates a successful forgery is $2^b$. In later sections, this thesis will advocate deriving multiple HMACs from the same cryptographic secret. To eliminate birthday paradox attacks on these truncated HMACs, it is essential that each HMAC key is used exactly once and that they are cryptographically independent.

In some contexts, the security requirement will still involve multiple spoofing attempts. For instance, for SBAS, the security requirement may include all possible attempts on a landing approach. To compute the security with $w$ attempts, this follows for a single HMAC key:

$$\Pr(\text{Forgery Success}) = 1 - (1 - 2^{-b})^w \leq w \cdot 2^{-b} \; . \tag{1.1}$$

### MAC Accumulation

When the security level of a truncated MAC is not sufficient, several references regarding Galileo suggest MAC Accumulation [41, 48, 49, 53]. With MAC Accumulation, the receiver must verify multiple purportedly authenticated messages together before using any of the information. When messages are authenticated with cryptographically independent MAC keys (as prescribed in Section 3.1), the probability that an adversary can spoof each of the MACs is independent. This means that the security level can be added. For instance, two 16-bit HMACs from [13] on the same information provide 32-bit security. MAC Accumulation can be useful for GNSS because often, the authenticated information remains constant over rigid intervals, changing at a strict cadence.

| **Type** | **Primitive** |
|---|---|
| ENC | AES [95] |
| H | SHA2 [103]<br>SHA3 [105]<br>KECCAK [101] |
| KDF | HMAC [101]<br>KMAC [101]<br>AES [95] |
| Symmetric Signature | HMAC [35, 99]<br>KMAC [104]<br>AES-GCM [100] |
| Asymmetric Signature | ECDSA [96]<br>EDDSA [96] |

Table 1.1: A List Of Standardized Cryptographic Primitives Appropriate For Any GNSS Application.

### 1.3.3 Recommended Primitives

Cryptographic agility refers to the ability to switch between cryptographic primitives. This is useful in case a cryptographic primitive is demonstrated to be broken (e.g., [106]) by enabling rapid adaptations without disruptive changes to the protocol. From the background of Section 1.3.1, a GNSS designer should incorporate as many functions from each category as possible. Table 1.1 provides a comprehensive list of the standard primitives for use with GNSS.

After great thought and care, a GNSS designer might consider replacing ECDSA with EC-Schnorr or utilizing ECHO as a hash function for hardware acceleration convenience [21, 90]. AES in Galois Counter Mode (AES-GCM) should not be used as a general substitute for HMAC, and the tags generated from AES-GCM should not be truncated below 96 bits [100].

I did not include any recommended quantum-resistant cryptographic primitives because the protocols under consideration are quickly changing; however, there are a few in Tables 4.1 and 4.2 for comparison purposes. Additional information on quantum resistance is in Section 1.3.7. If you are reading this resource looking for quantum-resistant protocols, you should seek another resource that is up to date.

### 1.3.4 TESLA

Timed Efficient Stream Loss-tolerant Authentication (TESLA) is an authentication protocol based on hiding bit commitments and preimage-resistant hash functions with several properties relevant to GNSS [80]. TESLA ***must*** be used together with traditional asymmetric digital signatures. However, TESLA enables massive bandwidth efficiency improvements and loss-tolerance over asymmetric-digital-signature-only protocols. This section will outline the basic protocol for TESLA.

TESLA will require generating an enormous number of what are traditionally called keys. For the intelligibility of this work, I cannot tolerate yet another object called a key. Moreover, in the context of this work being tailored to GNSS designers and other designers not as familiar with Cryptography, TESLA is better understood from a ***geometric*** perspective. In Section 1.3.5, I argue why one should adopt this terminology, but for now, I introduce TESLA with my terminology.

Let the set of all integers between 0 and $2^b - 1$ be called the hash point set, and each integer therein, a hash point ($p$). A hash point is simply an integer among $\mathbb{Z}_{2^b}$, but a hash point is either the output of a hash function or a random integer. A hash path is a collection of hash points related via repeated application of a hash function. Now let the hash path function (HPF) be a preimage-resistant function on the hash point set constructed with H, cryptographic salt, a counter $t$, and truncation:

$$\mathrm{HPF}(p, \mathrm{salt}, t) = \mathrm{TRUNC}\left( \mathrm{H}(p || \,\mathrm{salt}\, || t), b \right) \ . \tag{1.2}$$

In Eq. (1.2), the counter $t$ will be the time when hash point $p$ is applicable, but it does not have to be. Some salt and a counter is necessary to allow HPF to serve as a hiding commitment function [32, 71]. For now, I let the salt length be the same size as the hash point set. Since HPF includes a random salt and a cryptographic hash, HPF provides a hiding commitment with preimage resistance. To generate a hash path, one should draw a random hash point $p_0$ and a salt and then repeatedly apply

HPF the hash path length $L + 1$ times:

$$p_0 \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_{2^b} \tag{1.3}$$

$$\text{salt} \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_{2^b} \tag{1.4}$$

$$p_i = \mathrm{HPF}(p_{i-1}, \text{salt}, t_{i-1}) \quad \forall i \in \{1, ..., L\} \ . \tag{1.5}$$

To assist with the GNSS designer's intuition, I invoke a geometric interpretation: the ***one-way*** hash path of hash points in Fig. 1.4. After randomly selecting $p_0$, the rest of the hash path is deterministic but will appear as randomly bouncing on a number line of the 0 to $2^b - 1$ hash point set. Rather than representing them tangled on a number line, drawing them in order of computation will make sense, like in Fig. 1.5.

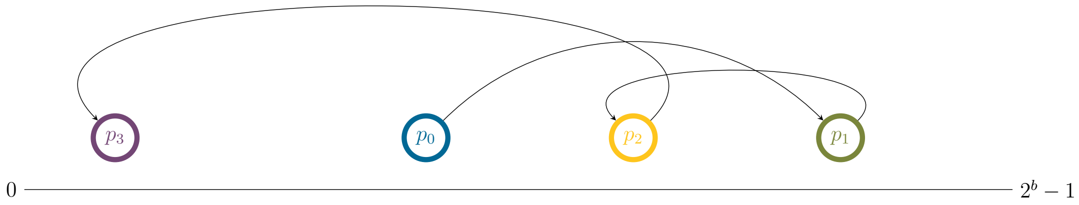


Figure 1.4: Conceptual Diagram Of A Hash Path Of Hash Points On A Number Line.

The diagram depicts a hash path of hash points placed on a number line representing the hash point set. Each arrow represents an HPF application. After randomly selecting $p_0$, the hash path is deterministic but will appear to randomly bounce on the number line of the hash point set.

TESLA is an authentication protocol to send messages from the provider to receivers. The provider will generate a hash path and hold all the hash points secret except the one computed last, $p_L$. $p_L$ is called the hash path end (HPE). The provider will broadcast them in ***reverse*** order. Hence, I now introduce the conceptual diagram of Fig. 1.6 depicted from left to right in order of ***release***. Diagrams similar to Fig. 1.6 will pervade this thesis. They will all read from left to right with increasing time of release by the provider or receipt by the receiver. The provider will broadcast the objects to the right later than those to the left. Arrows shall mean a one-way

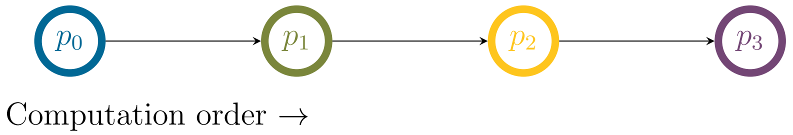


Figure 1.5: Conceptual Diagram Of Hash Path Depicted In Computation Order.

Each arrow represents an HPF application. After randomly selecting $p_0$, the hash path is deterministic but will appear to randomly bounce on the number line of the hash point set (not represented here, but represented in Fig. 1.4).

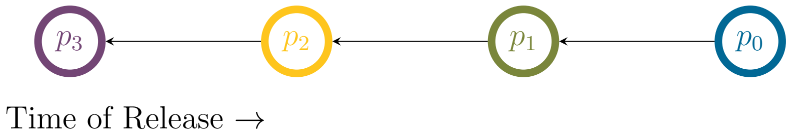


Figure 1.6: Conceptual Diagram Of Hash Path In Reverse Computation Order.

The diagram depicts the hash points placed in the ***reverse*** hash point generation order. The left-right direction now represents the time of broadcast of each hash point. Each arrow represents an HPF application. After randomly selecting $p_0$, the hash path is deterministic but will appear to randomly bounce on the number line of the hash point set (not represented here, but represented in Fig. 1.4).

function: either HPF, KDF, or CMF.

After generating the hash path, the GNSS provider must generate and distribute an asymmetric digital signature on $p_L$. To authenticate information, the GNSS provider will derive CMF keys using KDF from a ***secret, unreleased*** hash point. The derived keys will be used to authenticate a large number of objects using CMF. Then, later on a fixed schedule, the GNSS provider will stop using that hash point and broadcast it. After the hash point is broadcast, it is no longer secret and will not be used to generate CMTs. Instead, yet another preimage hash point will be used. Fig. 1.7 provides a conceptual diagram of this process.

The release of every hash point except $p_L$ must be on a rigid and known schedule. $p_L$ and its digital signature can be broadcast in a more flexible time frame that is better understood as TESLA ***maintenance*** data. For instance, when the GNSS provider knows it will ***soon*** run out of secret preimage hash points in the current

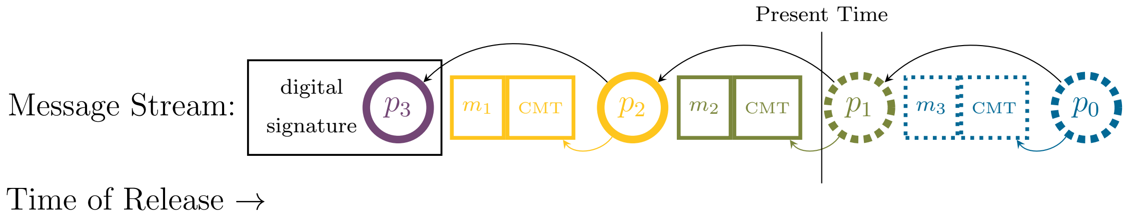


Figure 1.7: Conceptual Diagram Of A Hash Path For Authentication With TESLA.

The objects in the Message Stream to the left of the Present Time line have been broadcast. The dotted objects to the right of the Present Time line are only known to the GNSS provider because they have not yet been broadcast. The colors correspond to the information authenticated by a particular hash point. The GNSS provider must also generate and distribute a digital signature of $p_l$ ($p_3$ in the diagram). In the diagram, hash point $p_1$ will authenticate $m_2$ via the accompanying CMT. The hash point indexing is in the ***reverse*** order of the message indexing because the hash path must be released in reverse order. When the GNSS provider broadcasts $p_1$, the GNSS will cease using $p_1$ for authentication and instead will use the ***secret*** preimage $p_0$. Each black arrow represents an HPF application, which is a one-way function resulting from the underlying preimage-resistant hash function. Therefore, the adversary cannot compute $p_0$ or another preimage hash point of $p_1$. This repeats until the GNSS provider uses every secret preimage hash point, when it must generate another hash path.

hash path, it would be prudent to distribute the next hash path's $p_L$ and digital signature within the message stream to ensure a smooth transition. In other words, and concretely with Fig. 1.7, the $p_L$ and its digital signature will be distributed among the messages (e.g., within $m_1$ through $m_3$). But because the DS and HPE are released first, they appear on the far left of the diagram of Fig. 1.7. Understanding how to use a GNSS TESLA hash path should be studied separately from the associated TESLA maintenance (e.g., the DS and HPE, and salt) and will receive separate chapter treatment in this work. From now on, the digital signature may not be conceptually represented within the message stream. They can be entirely omitted or depicted in another row (e.g., Fig. 1.8).

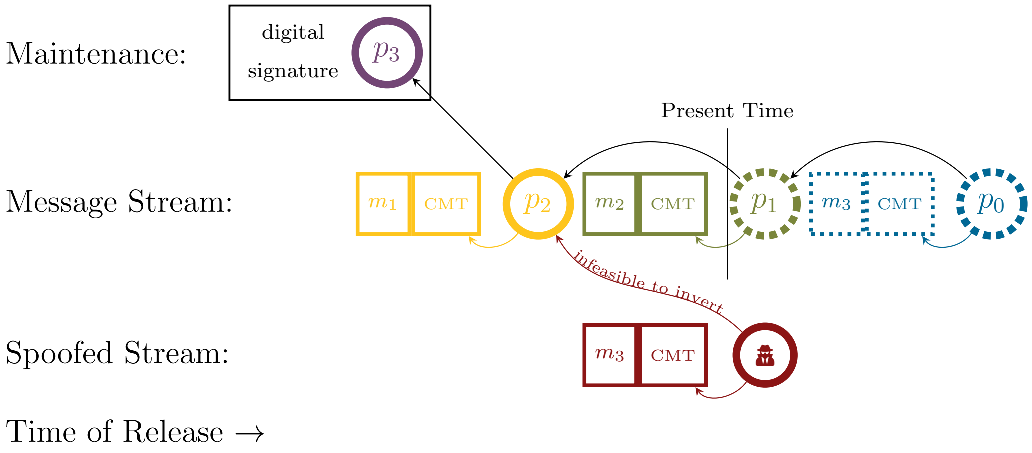


Figure 1.8: Conceptual Diagram Of The Authentication Security Argument For TESLA.

In the diagram, the receiver receives the authentic **green** message $m_2$ and its CMT generated with a secret hash point. The receiver then knows not to accept additional messages associated with a soon-to-be-released hash point $p_1$. Then, the GNSS provider broadcasts $p_1$. The receiver checks that repeated application of HPF on $p_1$ leads to a hash point with a valid digital signature. The receiver checks that $m_2$ and $p_1$ generate the receiver CMT. To fool a receiver into accepting a forged message, the adversary must compute a preimage of a hash point within the hash path before the authentic hash point is released. After breaking the preimage-property of the HPF, then the adversary can compute a forged message that meets the three security conditions within TESLA.

I now arrive at the final conceptual diagram of this section, Fig. 1.8. Fig. 1.8 includes all the pieces necessary to complete the TESLA authentication security argument (except the time synchronization issue introduced in Section 1.3.6). The receiver knows a message to be authentic when the following conditions are met:

(1) ***receipt safety***: the message and CMT arrived before the release of the associated hash point (Section 1.3.6),

(2) ***message authenticity***: repeated application of the HPF on the received hash point leads to an HPE signed with a valid DS, and

(3) ***message integrity***: the message and hash point derive the received CMT.

Because the authentic information must be in a stream, the receiver may only attempt

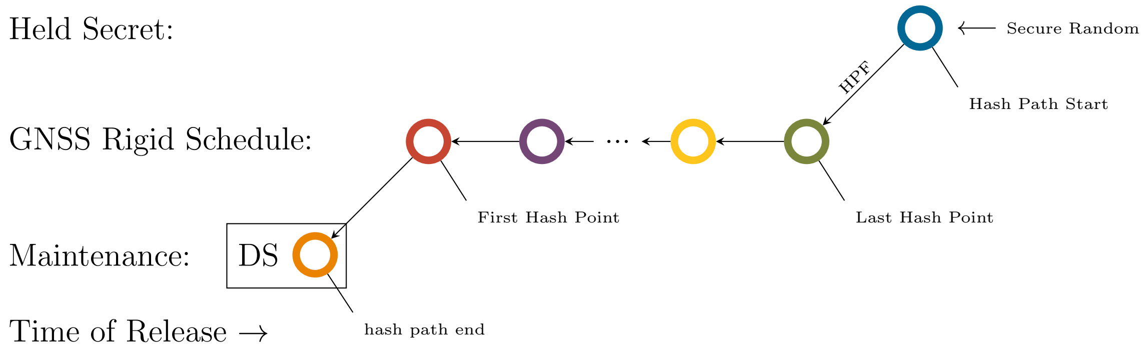


Figure 1.9: Conceptual Diagram With The Terminology Of The Anatomy Of A Hash Path.

> The geometry and temporal release of a hash path are in reverse of each other, causing a challenge with clear communication. In this work, I adopt separate terminology for geometry and temporary release: start/end and first/last, respectively. The GNSS designer will be more interested in the geometry; whereas, the receiver will be more interested in the release. Having the last hash point and hash path start be separate is not strictly necessary; however, separating them helps with clearer terminology, and it is a good idea to not allow the adversary direct access to the output of the constellation's secure random number generator.

to check the conditions of a single message for a single message slot ***once***.

Hash points are generated by the GNSS provider in ***reverse*** order of their release. With Section 1.3.5 and Chapter 3, I adopt geometric terminology to assist with design intuition. To avoid confusion, when speaking about the anatomy of a hash path, I will separate the geometric and temporal vocabulary with start/end and first/last, respectively. These are labeled in Fig. 1.9: hash path start, hash path end, first hash point, and last hash point. The geometric and the release-time considerations are explicitly separated for clarity.

The hash path start is a cryptographically secure random integer (from Eq. (1.4)). From the hash path start, the provider computes a large number of hash points to arrive as the HPE. All of the hash points are held secret, except for the HPE. From Fig. 1.9, the very first hash point revealed by the provider will be called the first hash point of a hash path. The very last hash point revealed by the provider will be called the last hash point. The hash path end is distributed separately via maintenance distributions of a particular hash path, whereas the first to the last hash points are revealed on a strict schedule. The first to the last hash point is used to authenticate messages in the message stream, whereas the HPE and its DS authenticates the hash path itself only.

#### Authenticated Encryption with TESLA

If a GNSS designer must encrypt the signal (e.g., to restrict access to paying subscribers), GNSS could utilize AES-GCM to provide authenticated encryption. However, the MAC should not be truncated to the level allowed for GNSS TESLA [100], limiting the applicability of AES-GCM use. Instead, the GNSS can utilize the existing TESLA protocol to authenticate encrypted messages, providing secrecy, integrity, and authenticity. Important to note: TESLA should only be used to authenticate ***directly*** on the ciphertext (i.e., Encrypt-then-MAC). Another construction should ***not*** be used (e.g., MAC the plain text and then encrypt —MAC-then-encrypt) because there are attacks known to break constructions that are ***not*** Encrypt-then-MAC [28].

### 1.3.5 Why use Hash Point Terminology?

My choice to adopt a new vocabulary for TESLA resulted from experience (1) learning how TESLA works, (2) communicating with others on how TESLA works, and (3) creating a design framework to leverage TESLA's advantages correctly. Many different and inconsistent terminologies exist in the literature, usually calling what I call hash points something related to locks or plants. The first problem with key is that there are already too many keys relevant to GNSS (e.g., public and private keys of multiple levels, encryption and decryption keys, CMF keys, and others), and as seen in later chapters, I will derive thousands of objects that could be called keys. The second problem with key is that there is no connection to the one-way and temporal relation of how TESLA uses them ephemerally. The primary issue with root or stem is the confusion resulting from the multiple apparent interpretations. When examining a plant's vascular system, plants branch up from the ground (e.g., stems) and down into the ground (e.g., roots). All TESLA geometry will branch in a single direction, or else it is insecure. Correctly leveraging TESLA's potential for GNSS requires understanding the geometry of one-way pseudorandom functions and constructing geometries along a ***time*** axis. Authentication breaks if the wrong object is broadcast at the wrong time.

The Git software version control system inspired this work's emphasis on geometry. Git uses a cryptographic hash function operation on (1) the existing code's current cryptographic label, (2) new code changes, and (3) other metadata to create new code version cryptographic labels. The more advanced commands within Git are better intuitively understood as geometric operations in the hash point set. Like with the diagrams in this thesis, Git branching geometry can get complicated. Unlike the diagrams in this thesis, the one-way cryptographic operations in Git version control diagrams point right in the direction of time (rather than left against time). To foreshadow Section 1.3.6, how far a single arrow points left against time will determine whether or not the GNSS protocol has authentication security. By utilizing geometric terminology with an interpretation directly related to authentication security, I hope the TESLA design process will be more straightforward and make finding errors obvious.

### 1.3.6 TESLA Loose-Time Synchronization

After a hash point is released, receivers must know not to accept authentication information (e.g., CMTs) derived from the revealed and now public hash point, lest they be susceptible to ***arbitrary*** forgery. Fig. 1.10 depicts how an adversary would attack a receiver not making this consideration.

TESLA counteracts the replay attack scenario of Fig. 1.10 by requiring an onboard clock with loose-time synchronization via a simple procedure that can accommodate any reasonable onboard clock [80]. Via the provided time synchronization procedure, the protocol introduces an ***individually-set*** delay between the message/CMT and the corresponding hash point. The worse the individual receiver clock, the longer the protocol-introduced delay within the individual's stream to accommodate.

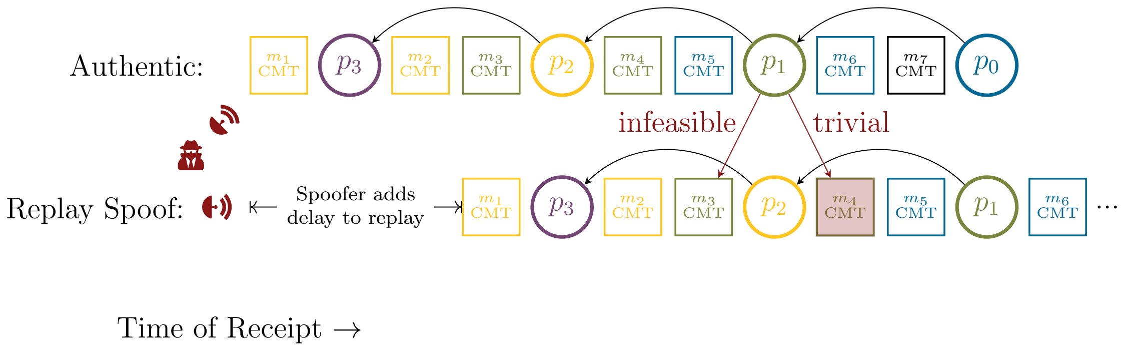


Figure 1.10: Conceptual Diagram Depicting How Replay Attacks Break TESLA Without Loose-time Synchronization.

The horizontal axis is the time of receipt of each object in the stream. The adversary receives the authentic message stream on time; whereas, the receiver receives the stream carried by the adversary with a delay. In this design, $p_1$ authenticates $m_3$ and $m_4$ (depicted in **green**). When the adversary delivers $m_3$ to the receiver, the adversary has not yet received $p_1$, so the adversary would successfully forge the $m_3$'s CMT at the rate of the CMT security level. When the adversary delivers $m_4$ to the receiver, the adversary has received $p_1$, so the adversary can use $p_1$ to forge $m_4$'s CMT. This diagram depicts an important geometric principle of the diagrams of this thesis: one-way operations are computationally infeasible by the adversary only if the arrow points left against time. An onboard clock must enforce that $m_3$ and $m_4$ are received before $p_1$ is released. TESLA loose-time synchronization describes the necessary accuracy and synchronization of the onboard clock to correctly enforce that $m_3$ and $m_4$ were received before $p_1$'s public broadcast.

GNSS TESLA introduces a profound change to the TESLA protocol: the delay must be the same for all users. With a constellation-wide delay, the loose-time synchronization requirement is turned on its head, requiring the receiver to enforce that authenticated information is received before the associated hash point, ***without*** relying on GNSS timing. Therefore, the receiver ***must*** have an onboard GNSS-independent clock (GIC) (for disconnected receivers, a real-time GIC) to authenticate with TESLA. Loose-time synchronization for GNSS TESLA means that

(1) the receiver asserts a correct bound on the uncertainty of its GIC in between synchronizations [45],

(2) the receiver utilizes that GIC to enforce that authenticated information (e.g.,

messages. CMTs) arrives before the release of the corresponding hash point, and

(3) the GIC is securely synchronized once its uncertainty bound exceeds a safety condition.

When a safe GIC determines that the message has arrived before the required time, the message has ***receipt safety*** (note the two other conditions from Section 1.3.4 needed for message authenticity and integrity). TESLA is not secure without loose-time synchronization, and a message is not authentic without receipt safety.

If GNSS seeks to authenticate its ranges, this loose-timing requirement is not inherent to TESLA. All authenticated ranging systems must have loose-time synchronization regardless of the underlying authentication protocol (discussed further in Section 2.1.2). This results from how the arrival time of the ranging system determines the Positioning, Navigation, and Timing (PNT) measurement. The adversary can observe and repeat the authentication cryptography within the ranging signal (though the adversary cannot predict that authentication cryptography before they observe it). The receiver must know that if an authenticated ranging signal arrives too late, then it must reject the ranging signal because it could be an adversarial replay.

This essential security condition can be confusing in the GNSS context because GNSS provides high-precision timing for receivers. The GIC does not replace GNSS timing; instead, the GIC provides a coarse time check. Except when resynchronizing with a two-way time synchronization server (e.g., when a network is available or during periodic maintenance), the GIC must be isolated from timing provided by any broadcast-only system like GNSS. It is ***never*** secure to adjust the GIC with GNSS unless the environment is separately monitored to ensure it is adequately free of spoofing signals.

### 1.3.7 Quantum Resistance

Current cryptographic security relies on arguments of computational work for classical computers. Quantum advantage refers to the moment when someone implements a quantum computer capable of performing additional tasks that a classical computer cannot complete. In the context of this work, a quantum computer will be able to execute Grover's search or Shor's period-finding algorithm [55, 93]. Should quantum advantage ever be achieved, to maintain equivalent quantum security to the pre-quantum classical security, the GNSS designer must double the hash point length $b$ and use the quantum-resistant public key authentication protocols from Tables 4.1 and 4.2 provided the underlying cryptographic primitives have not been broken by other means.

Grover's search enables the search of a black box database with a complexity of square root in the size of the database. This covers preimage resistance, where the adversary must search for a preimage of the hash function output. Concretely, the adversary should be able to find a 128-bit preimage hash point in 64-bit operation complexity and a 256-bit preimage hash point in 128-bit operation complexity. Therefore, to maintain the same security as before quantum advantage, the TESLA hash point length must double. Or, a 256-bit TESLA hash path is already quantum resistant.

Shor's algorithm enables the search for the period of a black box known to have a periodic output with polynomial complexity. This covers the hardness argument in deriving a public key from a private key, meaning that the adversary should be able to compute the private key from a public key, completely breaking all present public key authentication methods. No key lengthening can mitigate this risk because Shor's algorithm enables private key computation in polynomial time (e.g., the GNSS designer cannot simply double the strength of the certificate to accommodate). Rather, they must abandon current certificate types in favor of quantum-resistant ones. NIST is currently completing round four of its competition to standardize such a protocol [69], for which the finalists are listed in Tables 4.1 and 4.2.

The data bandwidth advantages discussed in Section 1.3.4 are especially important for quantum resistance. The present quantum public key infrastructure (PKI) under consideration by NIST requires enormous (by comparison to current PKI) certificates and signatures, whereas the TESLA mitigation needs only double the hash point length. In the later chapters, this thesis will separate the design of TESLA and the associated certificate distribution. In the quantum resistance context, this means that TESLA GNSS is better equipped. Quantum advantage would increase the amount of maintenance information distributed via the methods of Chapter 4, which only means more pages to distribute and a longer time to first authenticated fix (TFAF). And this would mean doubling the hash point length, which is the smallest component of time to authentication (TTA). Both of these changes are small compared to the alternative: a DS-only scheme that must completely abandon its cryptographic protocols with others with massive increases in data bandwidth.

## 1.4 GNSS Authentication

The literature has shown that cryptography can be useful in mitigating spoofing threats on GNSS data and ranging authentication.

This section discusses the basics of authentication in the GNSS context. Section 1.4.1 discusses an important performance measure for GNSS authentication designs. Section 1.4.2 discusses the prior art and challenges with data cryptographic authentication, whereas Section 1.4.3 discusses the prior art and challenges with ranging cryptographic authentication. Lastly, Section 1.4.4 discusses common GNSS authentication involving encryption.

The prior art of this section modifies existing GNSS systems in mid-to-geostationary orbit. Recently, there has been considerable interest in mega-constellations in low-earth orbit (LEO) capable of providing GNSS signals for greater accuracy, power, and data bandwidth. The concepts of this thesis will generally be applied to the existing systems because proposed LEO GNSS are so new and proprietary. However, this thesis will be in that context and serve as a guide to LEO GNSS security designers.

As discussed later, the challenges associated with cryptographic authentication motivate using TESLA for current system modifications. LEO GNSS constellations provide an opportunity to alleviate many of the challenges of current systems. While initially, this might lead to re-evaluating whether TESLA is necessary, I will make the case that TESLA will always be needed (see Section 2.1.2). Authentication will introduce a delay in the receiver's ability to utilize the PNT solution from the data. For private constellations, authentication takes away from revenue-generating bandwidth. The future space for competitive public and private innovation will be to minimize the TTA over the next few decades. In that context, the GNSS designer should always elect for the more efficient TESLA. The overall goal of this thesis is not to show that TESLA is needed; rather, to show how to leverage TESLA to the extent possible.

### 1.4.1 Time to Authentication

Like prior art, two performance indicators pervade this work: time to authentication and time to first authenticated fix [41, 52, 67, 70]. Suppose the receiver has all of the TESLA maintenance information (e.g., DS on the HPE) stored on board, which is the case for all receiver operations except during a cold start. TTA concerns itself with how long it takes the receiver to authenticate a message and CMT. TTA results from the accuracy of the onboard GNSS-independent clock and the underlying cryptographic-geometry construction: the subjects of Chapters 2 and 3, respectively. If the receiver initiates a cold start without any of the TESLA information, then TFAF is the primary performance indicator. TFAF concerns itself with how long a cold-start receiver needs to recover all of the TESLA maintenance information (i.e., all the information that supports the authentication of the current hash path). In the literature, authors sometimes refer to cold-start, warm-start, and hot-start receivers (or other taxonomies) depending on the extent to which TESLA maintenance information needs not be received, but these more specific taxonomies are not present in this work. TFAF is a one-time time cost when a receiver turns on, and its optimization involves designing the distribution of that maintenance information, the subject of Chapter 4. The TTA should be on the order of seconds, whereas the TFAF can take longer. These indicators are mutually exclusive on the receiver and receive separate, independent design treatment.

There are three components that make up the TTA: the interval of time for which a TESLA hash point applies ($I$), the TESLA commitment reveal delay ($\Theta$), and the

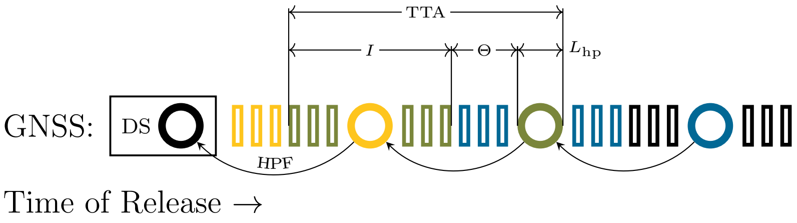


Figure 1.11: Conceptual Diagram Of The Components Of TTA.

The diagram depicts the three components that determine the TTA. The colors correspond to which data is associated with each hash point. $L_{\text{hp}}$ is the transmit time of the hash point, which is a function of the data rate of the channel. $\Theta$ is the time synchronization requirement, which is the subject of Chapter 2. $I$ is the interval length of a hash point's applicability to data in the stream, which is a function of how fast the applicable hash point to the message stream changes and the subject of Chapter 3. This diagram, and others like it, depicts the message time lengths shorter so that it can fit on the page. The order relation of the objects is more important to the purpose of this diagram, rather than showing accurate lengths for more objects. If the diagram were to show correct lengths for all objects, then $L_{\text{hp}}$ would be the shortest horizontal length in the diagram. $\Theta$ is dependent on how accommodating the GNSS constellation will be to receivers with inaccurate GICs.

length of time needed to transmit a hash point ($L_{\text{hp}}$). The interval can be bound as

$$\Theta + L_{\text{hp}} \leq TTA(m) \leq I + \Theta + L_{\text{hp}} = TTA\ . \tag{1.6}$$

The TTA($m$) of a specific message is a function of the message's relationship to the cadence of the hash point distribution. TTA shall mean the maximum of the interval to form a general performance indicator for a design, as in Eq. (1.6). Fig. 1.11 provides a conceptual diagram of that interval for an example geometry.

$I$, $\Theta$, and $L_{\text{hp}}$ are designed independently. $I$ is a function of the geometric features of the construction. Chapter 3 discusses how to construct geometries that shrink $I$. $\Theta$ is set to accommodate the accuracy of expected receiver clocks on the system. The GNSS designers can increase or decrease $\Theta$ to accommodate the expected receiver base by essentially shifting the interval $I$ left or right in Fig. 1.11. For instance, to accommodate worse clocks, Fig. 1.11 could be adjusted to have four messages (rather than three) between $I$ and the hash point. Chapter 2 discusses the implication with setting $\Theta$.

$L_{\mathrm{hp}}$ is a function of the desired TESLA security level and the system's data rate. Diagrams like Fig. 1.11 are principally meant to show the relation order of object release times. In Fig. 1.11, the message rectangles are shorter to represent an expanded time period on the page. However, in a real system with Fig. 1.11 drawn to scale horizontally, $L_{\mathrm{hp}}$ would be the shortest horizontal length. For instance, with SBAS with [13], messages are 250 bits, and a hash point would be 128-bits and transported within a 250-bit message.

In the near future, in the context of more frequent GNSS spoofing, systems will demand authenticated GNSS. Because ranging authentication requires a bit commitment protocol, authenticated GNSS requires there be a delay in PNT. After systems demand authenticated GNSS, they will demand GNSS authentication with a shorter TTA delay. Therefore, I predict that minimizing the TTA will be the principal concern of future authenticated GNSS constellations.

### 1.4.2 GNSS Navigation Message Authentication

The primary historical challenge with GNSS data authentication is transmitting the necessary cryptographic signature bits on existing low-data-bandwidth and lossy data channels. For instance, the Global Positioning System (GPS) C/A signal has a 50 bits per second (bps) data rate, and an ECDSA signature providing 128-bit security requires about 1024 bits. Sending one piece of information and requiring at least 20 seconds of messages containing the signature is impractical.

To accommodate the bandwidth issue, the GNSS community explored the lesser-utilized TESLA [19, 60]. At the time of writing, there are three potential practical applications of TESLA. The furthest along is Galileo's Open Service Navigation Message Authentication (OSNMA) (Section 1.4.2), which is currently broadcasting in the test phase and will start its operational phase imminently. After that is SBAS, which is commencing a global standardization process for a TESLA-based scheme (Section 1.4.2). Finally, there is Chimera (Section 1.4.2), which is close to launching a test satellite.

### Galileo's OSNMA

In the works for about a decade, Galileo's OSNMA is the first GNSS cryptographic authentication for a GNSS data channel [49, 53]. Galileo's OSNMA transmits authentication cryptography over the E1-B I/NAV message in a 40-bit field every two seconds [42, 67]. The data rate for E1-B is 125 bps. OSNMA's primary cryptographic primitives are TESLA with SHA256, HMAC-SHA256, and ECDSA [39]. Within the TESLA protocol, the HMAC tags are truncated on the order of 32 bits, depending on the operational mode. The TTA is on the order of two to five minutes, depending on the operational mode [53].

### SBAS NMA

SBAS is a group of constellations that broadcast GNSS corrections and integrity information over service volumes. Fixed ground stations within each service volume monitor GNSS and collect measurements, and the interpretations are forwarded to receivers via geostationary satellites. Among other requirements, SBAS must notify civilian aircraft within the service volume that GNSS is malfunctioning within six seconds to ensure safe GNSS-guided aviation. The GNSS community has been working on navigation message authentication (NMA) for the SBAS data [70].

SBAS has a 250 bps data bandwidth, which is much larger than the other current GNSS constellations. Presently, SBAS is capable of broadcasting on the in-phase (I) and quadrature (Q) but elects to only use the I for power reasons. Turning on the Q channel would take power away from the I channel, which would decrease the performance of SBAS at the service volume boundaries (e.g., Alaska). An SBAS TESLA design enables a TTA that meets the six-second alert requirement without using the Q-channel by efficiently utilizing unused space in the I-channel data bandwidth. The overwhelming majority of the TESLA design study for SBAS was completed by my predecessor [70]. Others have also considered the different SBAS authentication designs [50, 72]. This work will include several minor but essential enhancements to that TESLA design discussed in Sections 3.4 and 4.4. In addition, a significant component of this work is the TESLA ***maintenance***, which I explore through SBAS and go into detail in Chapter 4.

SBAS TESLA is presently undergoing standardization by the International Civil Aviation Organization (ICAO). The design has been modified over time to accommodate the breadth of stakeholders. The standardization effort is based on [13, 70], where a hash point is broadcast every six seconds. Each hash point derives five HMAC keys to compute five HMACs to sign five messages in a loss-tolerant way. Since then, more enhancements to the loss-tolerant aspects have been being studied, and a simpler digital structure is gaining favor. I refer to [13, 70] for greater details on the design selection, but I will discuss the maintenance topic in Chapter 4.

The scheme requires adding two message types (MTs) to the schedule. For L1, these are MT20 and MT21. For L5, these are MT50 and MT51.

MT20 and MT50 deliver the TESLA CMTs and delayed hash points to authenticate individual SBAS messages. MT20 and MT50 occur at a rigid frequency of one every six seconds, except during SBAS alerts. MT21 and MT51 deliver the TESLA maintenance information for the TESLA hash path. These messages fit in the remaining empty message slots of the SBAS message scheduler.

### Chimera NMA

The Air Force Research Laboratory, under its NTS-3 program, is pushing forward the Chimera protocol to address GNSS Authentication [1]. Chimera includes authentication for the data and the ranging of the L1C CNAV signal. This section discusses the data component NMA.

Chimera includes two independent authentication structures: the Slow Channel and the Fast Channel. The Slow Channel accommodates disconnected receivers that must use the GNSS data channel for authentication. The Fast Channel accommodates connected receivers that may receive authentication via a network connection. Fast-channel receivers can use their network connection to do NMA. Since internet authentication is ubiquitous and not under severe low-data-bandwidth challenges, its design falls outside the scope of this dissertation.

The Slow Channel NMA presents a more difficult challenge to Galileo's OSNMA due to L1C's smaller roughly 50 bps data rate. At the time of writing, the most recent

Chimera documentation delineates a new TESLA-based proposal [1, 2]. Previously, the NMA utilized ECDSA to generate digital signatures and distribute them within the available data channel bandwidth. Chimera's Slow Channel NMA will be around two to three minutes.

### 1.4.3 GNSS Ranging Authentication

GNSS ranging authentication utilizes hiding bit commitments and preimage-resistant cryptographic hash functions to derive pseudorandom ranging codes. The hidden commitment value derives a ***watermark*** in the ranging code or the ***entirety*** of the ranging code. A watermark induces perturbations in the ranging code, such as pseudorandom ranging code chip inversions or partial replacements with other cryptographic pseudorandom sequences (see Chapter 5). Noting that no other scheme has been proposed, I am reasonably confident there is no other possible ranging authentication mechanism beyond hiding commitments.

The following is a skeleton-ranging authentication procedure:

(1) the GNSS provider makes a hiding commitment where the revealed value will derive a watermark within the ranging code or the entire ranging code itself,

(2) the ranging signal is broadcast derived from the hidden commitment value,

(3) the receiver records the baseband ranging signal and stores it in a buffer,

(4) the receiver knows not to associate future ranging signals with the upcoming reveal of the hidden value that derives the perturbations or entire ranging code (i.e., time synchronization condition, see Section 1.3.6), and

(5) the provider reveals the committed value to the receiver to do additional signal processing described in Chapter 5.

This procedure is exactly the TESLA procedure, meaning that KDF can be used within the TESLA protocol to derive data-bandwidth efficient perturbations [7, 8]. In fact, under the right conditions, ranging authentication can be incorporated into a GNSS signal utilizing TESLA NMA with no additional data bandwidth.

Once the committed value is revealed by the GNSS provider, the receiver must perform additional signal processing. The additional signal processing involves correlation-like statistics discussed in Chapter 5. Since the commitment reveal must occur later, the receiver will need additional memory hardware over current receivers to enable authentication.

Since the arrival time of the signal determines the PNT, even when the signal includes a hidden commitment, the adversary can engage in Security Code Estimation and Replay (SCER) attacks or simply repeat the signal to enact a spoof.

In the following sections, I discuss a concept in the works for ranging authentication.

### Chimera Ranging Authentication

Section 1.4.2 discusses Chimera's data component NMA. This section discusses Chimera's ranging authentication. Chimera is the first proposal to introduce ranging authentication with watermarking under consideration for a GNSS constellation [1, 15, 56].

Chimera proposes two independent watermarks: one for the Slow Channel and the other for the Fast Channel. Ranging codes are split into sections, where some of the sections could include a Slow Channel perturbation, and the other sections could include a Fast Channel perturbation. The perturbed sections are pseudorandomly selected. If a section is perturbed, then the normal ranging code sequence is replaced with an AES-derived sequence. In [1], the perturbations derive from ECDSA signatures, so ECDSA provides the commitment protocol. Using ECDSA as the commitment protocol requires more data bandwidth than TESLA and introduces catastrophic loss-tolerance issues; however, a Chimera scheme with TESLA is underway.

With a network connection, fast-channel receivers can receive the information necessary to derive the perturbations in around five seconds. Given the comparative ease in designing authentication systems with a network connection, its consideration falls outside the scope of this dissertation. Without a network connection, slow-channel receivers will use the digital signatures contained within the data channel to derive the perturbations. Like with the Slow-Channel NMA, the TTA will be around two to three minutes.

### 1.4.4 GNSS Ranging PseudoAuthentication

GPS is, first and foremost, a military guidance system. Among its principal uses are weapons delivery and battlefield navigation. Any GNSS becomes a dangerous technology in the wrong hands. Hence, the United States millitary utilized encryption cryptography to ***restrict access*** to prohibit others from leveraging GPS to attack the United States and its allies. GPS includes several encrypted signals, and their sibling constellations followed suit (e.g., GPS's P(Y) code and M-Code, Galileo's Public Regulated Service).

A subset of privileged users have a symmetric encryption key to use an encrypted signal. The GNSS provider uses the encryption key and a cipher (e.g., AES) to encrypt the navigation message data and ranging codes. Anyone with the encryption key can utilize the signal or ***generate a spoofed signal***. Due to the limited distribution of the encryption key and assuming those with authorized access would never spoof, some claim that encrypted signals provide authentication. Encrypting a GNSS signal does not provide cryptographic authentication, despite claims to the contrary. While encrypted signals can make spoofing more complex, there is no pathway to provide a provable security claim.

If the encryption key is ever leaked, then anyone with the leaked key can generate a spoofed signal. To counteract this, present systems change decryption keys quickly and require arduous distribution procedures (e.g., requiring multiple personnel to distribute partial keys to battlefield receivers before use). Moreover, specialized hardware can make key extraction from privileged hardware more difficult. An authenticity argument requires that ***every*** other receiver be competent and honest, a feat too big to convince cryptographers of authenticity. However, assuming competence and honesty among privileged users can be sufficient for some applications.

### Galileo's ACAS

Galileo is currently developing Assisted Commercial Authentication Service (ACAS) [107]. ACAS uses the encrypted Public Regulated Service component to provide pseudoauthentication. Fragments of the encrypted Public Regulated Service ranging code are ***encrypted again*** with keys derived from the OSNMA TESLA hash path. The ciphertext resulting from the second encryption of the ranging code, called a Re-Encrypted Code Sequence (RECS), is stored on a server for download by the receiver. The receiver can download extended periods of RECSs for use in the future. As the receiver receives TESLA hash points, the receiver can decrypt the second encryption to compute the actual encrypted ranging code. Then, with the available encrypted ranging code, the receiver can process the encrypted ranging signal ***in the past***. If all privileged users were competent and honest, only GNSS could have generated this encrypted ranging signal, affording pseudoauthentication.

### Server Based Methods

Pseudoauthentication can be achieved with the assistance of a server (not colocated with the receiver) [62, 84]. The receiver records the baseband samples of where the encrypted signal should be and then sends the recording to the server. If the server is a privileged user with the encryption keys, it can easily process the receiver's recording and inform the receiver whether the encrypted signal is present as it should be. More advanced signal processing techniques enable the server to assist even without knowledge of the encrypted ranging code. Provided all users and the server are competent and honest, this concept can provide pseudoauthentication.

However, this concept requires sending high-frequency, incompressible baseband sample data from the receiver to the server, requiring a high-bandwidth internet connection (depending on the frequency and accuracy of the purported pseudoauthentication). Galileo's ACAS circumvents this issue by re-encrypting the encrypted signal for distribution in advance. Moreover, the server could simply distribute the encrypted ranging code ***after*** its use by GNSS over a server. Similarly, I suggest distributing the encryption keys ***after use*** to provide pseudoauthentication in Section 3.3.3.

## 1.5 Contributions

This dissertation is structured to provide a new treatment and comprehensive guide to "Designing Cryptography Systems for GNSS Data and Ranging Authentication" to contribute a "lasting value to the intellectual community"[54]. I address the present context of upcoming GNSS constellations without the severe bandwidth limitation, such as those in LEO or those that primarily serve connected receivers. Due to GNSS's data bandwidth limitation and cryptography's need for data, until now, ***cryptography dictated the product*** (i.e., the service the GNSS constellation could offer). With this guide, the GNSS designer should find that ***the product can dictate the cryptography***, alleviating the arduous data bandwidth demands imposed by cryptography and allowing the constellation to utilize the data bandwidth elsewhere or shrink the TTA. By focusing on cryptographic efficiency in the system design, there is less reliance on application-specific analysis to shave off bits from the authentication protocol [48, 70], which will assist with surviving the deprecation of future broken cryptographic primitives. Moreover, this guide's focus on reducing TTA should withstand the test of time, as I foresee TTA being a principal concern of GNSS innovation over the next century (see Section 1.4.1).

In addition, amid this guide, there are several "original constribution[s] to scholarship [and] scientific knowledge"[54], which I discuss in the following sections.

### Secure Time Synchronization with Broadcast-only TESLA and GNSS

My first contribution is the developments, derivations, and security proofs of the algorithms and protocols for a GNSS receiver to enforce receipt safety, which include

(1) the condition a receiver must enforce on the receiver-measured receipt time for all authenticated information, and together with a requirement on the GIC offset, proof of the condition's sufficiency for receipt safety;

(2) the condition a receiver must enforce on its GIC during network synchronization to ensure the offset requirement and determine when additional synchronization is necessary and proof that the same is sufficient for receipt safety; and

(3) the protocol a receiver must use to resynchronize its GIC safety, and proof that the protocol is sufficient for receipt safety.

For receivers unwilling to shut down when denied service during the protocols above, this thesis claims a security vulnerability within standard synchronization protocols and addresses it.

### Efficient TESLA Design Structures for GNSS

My second contribution is developing a new framework for efficient TESLA constructions that enable features without significant additional data bandwidth, including

(1) multi-cadence watermarking distribution strategies with a single signal watermark degradation;

(2) interleaving satellite watermarking authentication; and,

(3) additional third-party authentication and restricted access authentication.

The overall approach of this thesis's cryptographic construction enables cryptography-first GNSS authentication designs that shrink the TTA to the time synchronization limit, and this thesis claims several crypto-first signal designs. This thesis provides a framework so that the GNSS designer can utilize complex TESLA structures and features while ensuring necessary security requirements. Moreover, this thesis claims an SBAS TESLA scheme that can accommodate the SBAS alert requirement, salt, and multiple core-constellation ephemeris authentication.

### Efficient TESLA Maintenance Structures for GNSS

My third contribution is designing an SBAS over-the-air rekeying (OTAR) scheme enabling

(1) a cold-start TFAF of five minutes;

(2) multiple levels of certificates; and,

(3) various certificate expiration and revocation policies.

This thesis provides a framework so that the GNSS designer can design efficient TESLA maintenance structures in support of a TESLA scheme.

### Combinatorial Watermarking Functions for GNSS

My fourth contribution is creating a secure Combinatorial Watermarking Function that

(1) induces a constant degradation within each watermarked ranging code;

(2) induces degradation uniformly throughout each watermark; and,

(3) ensures that observation of the watermark reveals nothing about the watermarking seed from which the watermark derives.

The construction of this watermark is flexible and requires no additional data bandwidth, enabling application broadly to any existing or future GNSS authentication scheme.

### Distributions of Watermark-induced Receiver Statistics

My fifth contribution is deriving the security analysis of the Combinatorial Watermark and results from their mathematical construction enabling mathematically simple derivations, including

(1) the distribution of combinatorial-watermark-induced receiver statistics in the presence of Non-SCER and limited SCER adversaries;

(2) a dual receiver statistic paradigm that enables concise mathematical derivations, direct relations to adversary capability, and optimized watermark designs; and,

(3) two SCER attack strategies, including one incorporating soft information to vary chip power for better spoofing performance.

This thesis provides a framework so that a GNSS designer can design a watermark to meet specific security and false alarm requirements, minimize degradation and receiver memory requirements, specify how well an adversary must perform to break the watermark, and specify the required receiver signal processing.

### 1.5.1 Publications

This work is based on several peer-reviewed publications of which I am a primary or secondary author. These publications are listed below.

#### Conference Papers

- Todd Walter, Jason Anderson, and Sherman Lo. "SBAS Message Schemes to Support Inline Message Authentication". In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021).* 2021, pp. 474–484. DOI: 10.33012/2021.17908
- Ignacio Fernandez-Hernandez et al. "Message Authentication Candidates for the SBAS Dual Frequency Multi-Constellation Standard". In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021).* 2021, pp. 443–452. DOI: 10.33012/2021.17892
- Ignacio Fernández-Hernández et al. "SBAS Message Authentication: A Review of Protocols, Figures of Merit and Standardization Plans". In: *Proceedings of the 2021 International Technical Meeting of The Institute of Navigation.* 2021, pp. 111–124. DOI: 10.33012/2021.17829
- Brady O'Hanlon et al. "SBAS Signal Authentication". In: *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022).* 2022, pp. 3369–3377. DOI: 10.33012/2022.18443

- Jason Anderson, Sherman Lo, and Todd Walter. "Authentication Security of Combinatorial Watermarking for GNSS Signal Authentication". In: *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023).* 2023, pp. 495–509

- Todd Walter, Jason Anderson, and Sherman Lo. "Implementation of Data Authentication on SBAS". in: *Proceedings of the ION 2024 Pacific PNT Meeting.* 2024, pp. 709–721. DOI: 10.33012/2024.19631

**Peer-reviewed Conference Papers**

- Jason Anderson et al. "On SBAS Authentication with OTAR Schemes". In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021).* 2021, pp. 4288–4304

- Jason Anderson, Sherman Lo, and Todd Walter. "Efficient and Secure Use of Cryptography for Watermarked Signal Authentication". In: *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation.* 2022, pp. 68–82. DOI: 10.33012/2022.18228

- Jason Anderson, Sherman Lo, and Todd Walter. "Cryptographic Ranging Authentication with TESLA, Rapid Re-keying, and a PRF". in: *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation.* 2022, pp. 43–55. DOI: 10.33012/2022.18226

- Jason Anderson, Sherman Lo, and Todd Walter. "Time Synchronization for TESLA-based GNSS-enabled Systems". In: *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022).* 2022, pp. 3408–3417. DOI: 10.33012/2022.18442

- Jason Anderson, Sherman Lo, and Todd Walter. "Addressing a Critical Vulnerability in Upcoming Broadcast-only TESLA-based GNSS-enabled Systems". In: *Proceedings of the 2023 International Technical Meeting of The Institute of Navigation.* 2023, pp. 277–285. DOI: 10.33012/2023.18623

- Jason Anderson, Sherman Lo, and Todd Walter. "Combinatorial Watermarking for GNSS Signal Authentication". In: *Proceedings of the 2024 International Technical Meeting of The Institute of Navigation.* 2024, p. 314159. DOI: 10.33012/2024.19483

- Jason Anderson, Sherman Lo, and Todd Walter. "Revisiting Combinatorial Watermarking under SCER Adversarial Models". In: *Proceedings of the ION 2024 Pacific PNT Meeting.* 2024, pp. 732–744. DOI: 10.33012/2024.19633

**Peer-reviewed Journal Papers**

- Jason Anderson et al. "Authentication of Satellite-Based Augmentation Systems with Over-the-Air Rekeying Schemes". In: *NAVIGATION: Journal of the Institute of Navigation* 70.3 (2023). ISSN: 0028-1522. DOI: 10.33012/navi.595

- Jason Anderson, Sherman Lo, and Todd Walter. "Authentication Security of Combinatorial Watermarking for GNSS Signal Authentication". In: *NAVIGATION: Journal of the Institute of Navigation* 71.3 (2024). ISSN: 0028-1522. DOI: 10.33012/navi.655

- Jason Anderson, Sherman Lo, and Todd Walter. "Time Synchronization of TESLA-enabled GNSS Receivers". In: *IEEE Transactions on Aerospace and Electronic Systems* (Accepted, pending revisions 2025)

- Jason Anderson, Sherman Lo, and Todd Walter. "Revisiting Combinatorial Watermarking under SCER Adversarial Models". In: *NAVIGATION: Journal of the Institute of Navigation* (Accepted, pending revisions 2025)

# Chapter 2

# Broadcast-only TESLA Time Synchronization

> The only way out is through.
>
> Robert Frost

This chapter is about the necessary time synchronization protocols and timing checks on ***every*** authenticated information received within a Global Navigation Satellite System (GNSS) Timed Efficient Stream Loss-tolerant Authentication (TESLA) protocol[1]. The material of this chapter cannot be ignored, lest the receiver does not have authentication. Every receiver must have an onboard GNSS-independent clock (GIC) to assert Positioning, Navigation, and Timing (PNT) authentication, and the better the GICs among the constellation's user base, the better the time to authentication (TTA) achievable. As an easy reference, Section 2.3 provides the final procedures derived and discussed in the later sections of this chapter.

In my experience within the literature and navigation community, the topic of TESLA time synchronization is often left as an afterthought or ignored. It will be the most relevant and annoying design consideration for future GNSS authentication. It was a specific and intentional decision for this chapter to be first. Read it.

[1] This chapter is based on my three publications regarding time synchronization for broadcast-only TESLA for GNSS [3, 11, 12].

As discussed in Section 1.3.6, the broadcast-only nature of GNSS necessitates that receivers enforce a condition on the accuracy of its GIC and use its GIC to enforce that every piece of TESLA-authenticated information arrives at the receiver before the release of the corresponding hash point. If the receiver does not have a GIC, it will not be able to reject forged signals from replay attacks (Section 2.1.1). If the receiver's GIC does not meet the conditions of this chapter, it will not be able to reject forged signals from replay attacks. Section 2.1 discusses why these requirements are necessary, and Section 2.2 discusses how to select the time synchronization requirement based on the available clock.

In addition to the receiver's requirements regarding its GIC, the GNSS constellation must maintain time synchronization infrastructure for receivers as their GICs need resynchronization. Fig. 2.1 provides a conceptual diagram of the various aspects of this infrastructure and how it will be used. In the future, GNSS receivers, particularly those connected with the safety of life, will want (or be required) to enforce the authentication of current GNSS signals and future GNSS signals. Receivers might have continuous access to an internet connection (e.g., autonomous cars) or not (e.g., aircraft). Receivers may have access to GICs of varying drift rates. For instance, I expect receivers without an internet connection will have access to a low-drift GIC and need to conduct recurring clock checks during regular maintenance. In contrast, receivers with an internet connection can frequently check their low-cost GIC. Because this chapter's adversary model includes the ability to repeat and delay GNSS signals, it is reasonable to assume that the adversary can observe and delay network signals. Sections 2.5 and 2.6 discuss the needed modifications to network synchronization protocols for GNSS.

Cryptography does not provide protection against delays [16]. Unfortunately, for PNT authentication, the receipt time of a GNSS signal determines the receiver-deduced position and time. As discussed in the cat and mouse game from Section 1.2, an adversary could theoretically make very small (i.e., on the order of nanoseconds) delays on GNSS ranging signals to spoof a receiver's PNT. From the provable PNT authentication perspective, each nanosecond delay enables about three meters of uncertainty. This vulnerability lends to a Heisenburg-Uncertainty-Esk principle: to

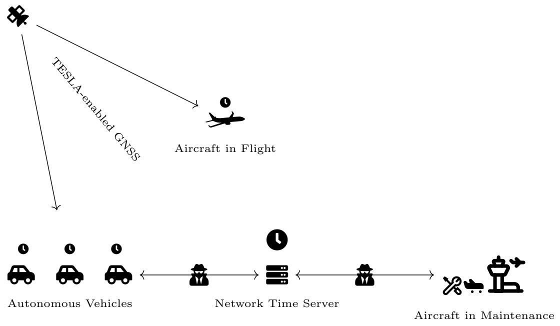


Figure 2.1: Conceptual Diagram Of Network Time Synchronization Infrastructure For GNSS TESLA.

In the future, receivers enforcing GNSS authentication will require additional network-based timing infrastructure due to the broadcast-only nature of GNSS. Some receivers, such as autonomous cars, will have continuous access to a network time server, allowing continuous checking to enforce the TESLA loose-time synchronization requirement. Other receivers, such as aircraft, might not have continuous access to a server but will use periodic maintenance and low-drift onboard clocks between maintenance sessions to enforce the TESLA loose-time synchronization requirement. An adversary can delay the receipt times of synchronization signals to and from a synchronization server.

prove your position, you must have perfect timing and know exactly when to expect the arrival of the GNSS ranging signal. But that should not mean giving up; rather, the GNSS can make these attacks very difficult (just not in the cryptographic sense).

## 2.1 GIC Threat Models

In this chapter, I will frequently describe the time of an event in two different clock frames: (1) the GNSS provider time and (2) the GIC time of a particular GNSS receiver attempting to assert the authenticity of a TESLA-enabled GNSS signal. Let a particular synchronization of a GIC be indexed by $l$, and let each synchronization include several sub-events indexed by $i$. An event $i$ during synchronization $l$ occurs at $t_i^l$ in the GNSS provider clock frame and $\tau_i^l$ in the receiver GIC frame, via the measurement Eq. (2.1) with $\theta^l$ as the offset of the GIC during synchronization $l$:

$$\tau_i^l = t_i^l + \theta^l \ . \tag{2.1}$$

During $l$, I reasonably assume the clock offset of the GIC ($\theta$) is constant, so I will frequently drop the superscript $l$. In Eq. (2.1), $\theta > 0$ means the receiver clock runs ***ahead*** of the provider time. Authentication security will break when the receiver clock excessively ***lags*** provider time, and the procedures in this chapter will provably determine or fix the health of a GIC.

### 2.1.1 Replay Attacks on GNSS TESLA NMA

The receipt time of a GNSS signal determines the receiver-deduced position and time. If a man-in-the-middle repeats the GNSS signal to a receiver, the receiver will deduce a time estimate that lags behind authentic GNSS time. For instance, a spoofer could delay the GNSS signal, slowly increasing the delay until the receiver accepts a spoofed message under TESLA message authenticity and message integrity. Here lies the Catch-22 that necessitates non-GNSS timing assistance for GNSS authentication security. Because of the broadcast-only nature of GNSS and how simply repeating a GNSS signal could be accepted by the receiver as a lagging signal, synchronization requires a non-GNSS, two-way, and recurring connection [68]. Because continuous two-synchronization is not feasible, the receiver must have an onboard GNSS-independent clock. And because receivers will have uncertainty with their GIC, the constellation must specify the TESLA commitment reveal delay ($\Theta$).

Fig. 1.10 within Section 1.3.6 shows the authentic and arbitrary forgery case within a replay attack. However, there is a spectrum of effectiveness for replay attacks as the adversary-induced delay increases. Fig. 2.2 provides conceptual diagrams of this spectrum. Within Fig. 2.2, Spoof A has the smallest delay, and Spoof D has the largest delay. An adversary could slowly increase the delay over the spoof to avoid detection, transitioning from Spoof A to Spoof D.

Recall that $\Theta$ is the length of time from (1) the release of the last bit of the latest authenticated information (e.g., messages and CMTs) to (2) the release of the first bit of the associated hash point. $\Theta$ will be selected to accommodate the clocks from Section 2.2.3 and will have the implication of Section 2.2.1. In Fig. 2.2, Spoof A has an adversarial delay of exactly $\Theta$. In Spoof A, since no information is received after

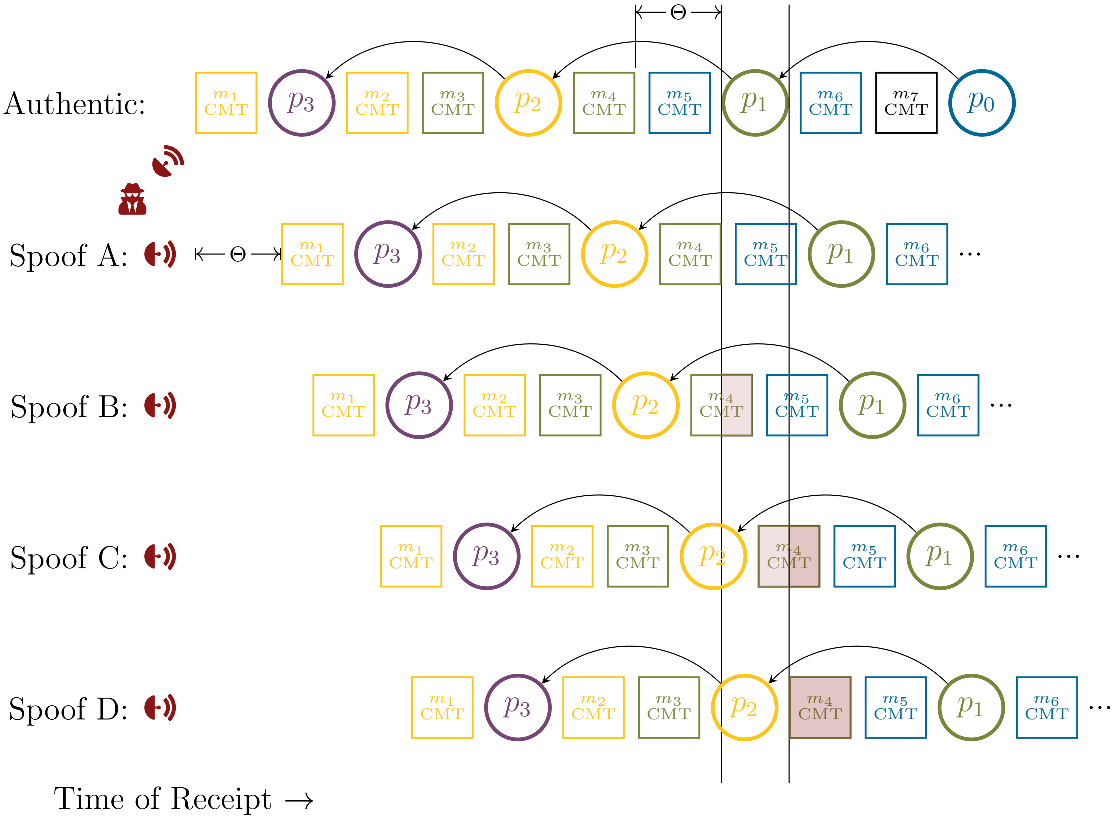


Figure 2.2: Conceptual Diagram Comparing The Efficacy Of Replay Attacks Of Varying Delays.

The horizontal axis is the time of receipt by the receiver for four spoofing attacks of differing delays with the adversary and receivers not under attack receiving the stream perfectly on time in the top row. The adversary observes then replays and modifies the authentic signal with a delay and attempts to spoof $m_3$ by generating a forged commitment-MAC tag (CMT). To counteract each of these spoofs, the receiver must use its safe GIC on each message. In Spoof A, the adversary delays the signal by exactly Θ, and the receiver receives the entire message $m_3$ and CMT before any of the corresponding hash point $p_1$ is publicly known. Even with the delay of Spoof A, $m_3$ is protected by TESLA. In Spoof B, the adversary delays the signal by slightly more than Θ where the adversary knows part of $p_1$ during the attempted message and CMT latter-portion forgery (light **cardinal**). Assuming effortless receiver-to-transmitting transport and negligible computation time, each known bit of $p_1$ erodes the afforded security. In Spoof C, some of the message-CMT bits arrive after the entire $p_1$ (**cardinal**), meaning the adversary can commence nearly an arbitrary forgery (the forgery must be consistent with the adversary-provided light-red bits). In Spoof D, the adversary knows $p_1$ before the scheduled $m_4$ transmission time and can commence an ***arbitrary*** forgery.

the distribution of the associated hash point, the full security level applies to that information (subject to Section 1.2).

In Spoof B, the adversary delays the signal beyond $\Theta$. Using the methods of this chapter, the GIC will reject messages in that scenario (and so on with Spoofs C and D). Recall that the adversary has zero latency, meaning it can instantaneously observe, process, and transmit (subject to cryptographic security). In Spoof B, the adversary knows part of the released hash point when it distributes its forged message and CMT to the receiver. As the delay increases, the zero-latency adversary can access more of the hash point, decreasing the effective security. For instance, if the adversary gets to know 50 bits of a 128-bit hash point before transmitting a spoofed message, then the security of the TESLA decreases to 78 bits. At this point, the adversary may only make an exhaustive guess of the hash point when generating a message and CMT pair.

In Spoof C, part of the authentication information comes after the complete distribution of the hash point. The lighter **cardinal** section comes before, and the darker **cardinal** section comes after. Whereas, with the lighter **cardinal** section, the adversary must make an exhaustive guess on those bits, with the darker **cardinal** section, the adversary can use the full hash point to compute an arbitrary forgery on those bits. In Spoof C, the adversary must make a commitment guess with the first part but has the free ability to forge the last part. In Spoof D, the entire hash point is known to the adversary to generate its forgery, meaning the adversary can forge a complete message and CMT without regard to the commitment property of the CMT.

It would certainly be wise to use the GIC to detect jumps in the delays of the signal. For this chapter, the adversary can evade such checks. Instead, this chapter concerns itself with bounds on the GIC accuracy and provably secure checks on the authenticated information.

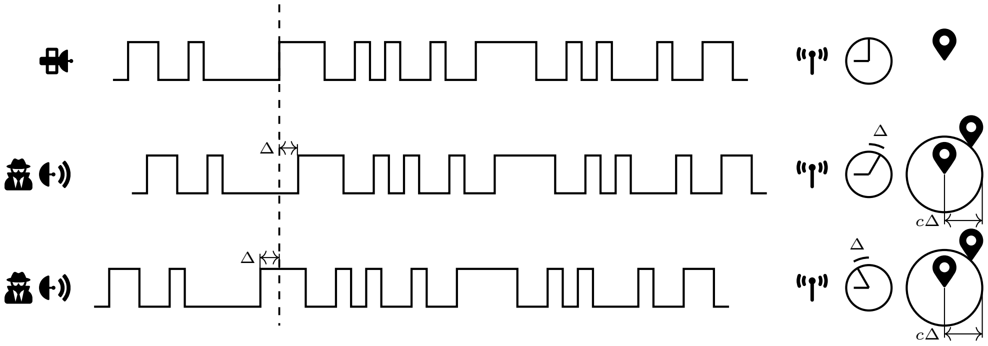


Figure 2.3: Conceptual Diagram Of The Spoofing Effects Of Delayed Ranging Signals.

The top row contains the authentic ranging code traversing from the GNSS satellite to the receiver. In the middle row, the adversary causes the ranging code to arrive ***early*** to the receiver. In the bottom row, the adversary causes the ranging code to arrive ***late*** to the receiver. In no case does the adversary change the ranging code itself. The $\Delta$ induced by the signal can cause the receiver-deduced timing to change by $\Delta$ if the adversary induces the same $\Delta$ for all GNSS satellites. If the adversary induces different $\Delta$ among all the GNSS satellites, the deduced PNT solution can be manipulated in any direction up to $c\Delta t$, where $c$ is the speed of light.

### 2.1.2 Delay Attacks on GNSS TESLA Ranging

The receipt time of a GNSS signal determines the receiver-deduced position and time. In the cat and mouse game from Section 1.2, some adversaries will have the equipment and know how to freely manipulate the receiver's PNT solution. Unfortunately, the adversary only needs to manipulate the signal with delays to achieve its aims; however, one can make these attacks substantially more difficult with the methods of this thesis.

Fig. 2.3 provides a conceptual diagram of how adversary-induced delays affect the receiver-deduced PNT solution. Depending on the adversary-induced delay $\Delta$ for each of the GNSS satellites in the constellation, the receiver-deduced PNT solution can be manipulated by up to $\Delta$ in time and up to $c\Delta$ in position (where $c$ is the speed of light). Because $c$ is so large, small perturbations can drastically affect the received-deduced position.

In Chapter 3, I will discuss how to generate pseudorandom ranging codes within the TESLA framework. Within the TESLA framework, these ranging codes can derive from hidden preimage hash points (to be released later), meaning the adversary will

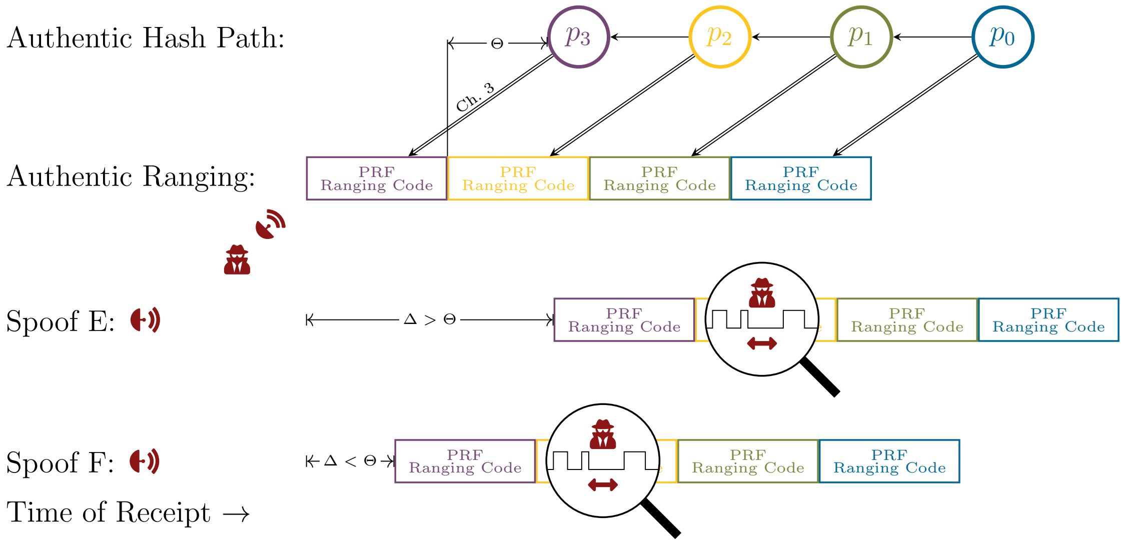


Figure 2.4: Spoofing With Ranging Code Delays And Θ.

The GNSS signal generates pseudorandom function (PRF) ranging codes from a hash path via the TESLA framework (see Chapter 3), meaning that the ranging code is not known to the adversary at broadcast. The ranging code derives from a hash point distributed later according to the TESLA protocol's Θ. In Spoof E, the adversary waits until the distribution of the hash point, derives the hidden ranging code, and then spoofs a ranging signal with a delay larger than Θ. Within the magnifying glass, the adversary is able to make small delay manipulations to affect the receiver-deduced PNT solution exactly as depicted in Fig. 2.3. Because the delay exceeds Θ, Spoof E will be rejected by the GIC. In Spoof F, the adversary delay is less than Θ, so it will ***not*** be rejected by the GIC. In Spoof F, without knowledge of the hash point, the adversary must listen to (or estimate) the PRF ranging code and then deliver it to the receiver. The complexity of having to listen/estimate and then spoof the PRF ranging code quickly (i.e., with a delay less than Θ) makes Spoof F materially more difficult than Spoof E.

not know what the ranging code is when it is broadcast. However, the delays in Fig. 2.3 are of the entire signal itself, meaning such an attack is possible without knowing the pseudorandom ranging code. Hiding the ranging code adds a material complexity to the adversary's delay attack: the adversary will need to estimate each ranging code chip and then deliver them to the victim receiver. However, this added complexity only applies to delays less than Θ, which is the subject of Fig. 2.4.

Fig. 2.4 compares two spoofs utilizing the methods from Fig. 2.3. Spoof E has a total delay greater than $\Theta$, and Spoof F has a total delay less than $\Theta$. Spoof E will be rejected by the GIC, but Spoof F will not; however, Spoof F is remarkably more complex than Spoof E. With Spoof E, the adversary receives the hash point and then generates a consistent signal with a delay greater than $\Theta$. Spoof F is more complex because the spoofer does not know the PRF ranging code when it generates the spoofed ranging code. Instead, it must listen to the signal (or estimate the ranging code one chip at a time) and deliver the delayed signal to the receiver. The PRF signal is not compressible, and the signal processing requires advanced technical know-how.

Ranging signals are usually low power, with the ranging code length enabling receivers to lift the signal out of the noise. Without knowledge of the code, the adversary needs sophisticated radio equipment. To avoid detection by the GIC, the adversary must observe, process, and deliver the signal to the receiver with a delay less than $\Theta$. As $\Theta \to 0$, this attack is more difficult. Because of the fundamental nature of the speed of light, this requires the spoofer to be nearer to the victim, have faster computational processing, and be physically smaller (so as to not be noticeable by the victim or authorities).

### GNSS Authentication Needs TESLA

TESLA is a lesser-known authentication protocol (e.g., compared to public key protocols like the Elliptic Curve Digital Signature Algorithm (ECDSA)). TESLA must use an asymmetric digital signature (DS) protocol anyway, and TESLA introduces the time synchronization requirement studied in this chapter. Given the additional complexity, why bother using TESLA over DS? Chapter 3 discusses the incredible bandwidth savings or TTA improvement, but suppose a GNSS designer ignores these advantages. Instead, they focus on the arduous efforts required to account for the time synchronization requirement. Why should that GNSS designer still use TESLA for GNSS?

They should do so because the time synchronization requirement is fundamental to the cryptographic authentication of ranges. Regardless of whether a GNSS utilizes TESLA for ranging authentication or not, the TESLA loose-time synchronization requirement is needed for authentication security.

Suppose the PRF ranging code was generated from a DS protocol, such as ECDSA. The method to generate the signature is one-way, providing equivalent protection as the preimage resistance of the hash function within the hash path. Instead of having each of the hash points relate via the hash path function (HPF), they are generated via a secret certificate key that is never released. But DS protocols do not address the attack from Fig. 2.4, meaning that the time synchronization issues of this chapter are still needed.

TESLA poses the additional advantage of explicitly addressing the time synchronization needed for authentication security. Even when a GNSS designer uses DSs directly for the PRF ranging codes and does not use TESLA for the cryptography in GNSS authentication protocol, that GNSS designer must still adapt the TESLA loose-time synchronization requirement to the GNSS protocol. So if the time synchronization requirement is not avoidable, then it is best to utilize TESLA because it is designed explicitly for this delayed-authentication problem and provides the other advantages from Chapter 3.

### 2.1.3 Delay Attacks on NTS

Network Time Protocol (NTP) from [64] provides a simple synchronization protocol between two clocks with a network connection. Immediately, I use its security extension, Network Time Security (NTS), which is NTP augmented with authentication cryptography [51]. I provide a brief algorithmic overview of NTS in Algorithm 2.1. The methods within this chapter can immediately extend to PTP [59]. I use NTS herein for simplicity and brevity. GNSS serves as an excellent proxy for reference time. NTP Stage 0 clocks are usually atomic clocks or GNSS (noting that GNSS is a collection of orbiting atomic clocks). UTC-GPS is tightly constrained to UTC-USNO, so GNSS is the reference time for this work.

NTS provides authentication security on the ***content*** of the messages [51]. However, the ***measured*** sending and arrival times determine the estimated clock drift, and the adversary can manipulate the measured arrival times via man-in-the-middle delays (see Fig. 2.18, though the defined variables are necessary for the discussion

**Algorithm 2.1:** Network Time Security (DO NOT USE FOR GNSS TESLA).

```
1 GNSS provider (or a delegate) and receiver establish an asymmetric
    authentication security instance.
2 Receiver draws a nonce $\eta$.
3 Receiver sends message $m_1 = (\eta, \tau_1, s_1^{\text{receiver}})$ where $\tau_1$ is the time receiver
    recorded at the moment of sending $m_1$ and $s_1^{\text{receiver}}$ is receiver's
    authentication signature on $(\eta, \tau_1)$.
4 Provider records $t_2$, the time of receipt of the message $m_1$.
5 Provider sends message $m_2 = (\eta, \tau_1, t_2, t_3, s_2^{\text{provider}})$ back to receiver, where
    $s_2^{\text{provider}}$ is provider's authentication signature on $(\eta, \tau_1, t_2, t_3)$. $t_3$ is the
    moment that $m_2$ is transmitted back to the receiver.
6 Receiver records $\tau_4$ at the moment of receipt of message $m_2$.
7 The measured clock drift is $\hat{\theta} = \frac{1}{2}(\tau_1 - t_2 - t_3 + \tau_4)$ assuming that the transit
    time of $m_1$ and $m_2$ are the same.
```

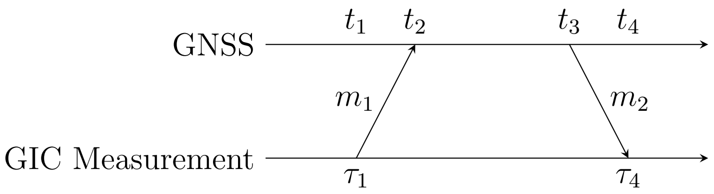


Figure 2.5: Conceptual diagram of Network Time Security.

A conceptual diagram of Algorithm 2.1. The diagram depicts increasing time from left to right in the provider and receiver clock frames, and the messages shared between them. The protocol presumes provider and receiver have already established a cryptographic authentication instance to protect the ***content*** of the messages transmitted. The protocol is susceptible to adversary-induced delays in message transmission (like in Fig. 2.18).

of this section). No cryptographic protection is available for message delays [16]. There are several suggested mitigations to man-in-the-middle delays. These include querying multiple random NTS servers and rejecting synchronization with large NTS round-trip times. While helpful, they do not provide provable safety beyond the round-trip time uncertainty, and I am primarily concerned about ***lagging*** clocks in this work.

NTS uses two messages: (1) a message from the receiver to the provider and (2) a message from the provider to the receiver. The times of sending and receipt of the two messages are referred to as synchronization events 1, 2, 3, and 4, respectively, and determine the estimated receiver clock offset $\hat{\theta}$. These events are diagrammed in Fig. 2.5. The estimated $\hat{\theta}$ derives from the assumption that the receiver-to-provider and the provider-to-receiver message ping transit time are the same, as in Eq. (2.2) to Eq. (2.3) via the measurement Eq. (2.1):

$$\begin{aligned} t_2^l - t_1^l &= t_4^l - t_3^l && (2.2)\\ t_2^l - (\tau_1^l - \hat{\theta}^l) &= (\tau_4^l - \hat{\theta}^l) - t_3^l \\ \hat{\theta}^l &= \frac{1}{2}(\tau_1^l - t_2^l - t_3^l + \tau_4^l) \\ \hat{\theta}^l &= \frac{1}{2}(\tau_1^l + \tau_4^l) - \frac{1}{2}(t_2^l + t_3^l) \ . && (2.3) \end{aligned}$$

Equation (2.3) is arranged in the form that corresponds to estimating the horizontal midpoint of the trapezoid from Figure 2.5.

While an adversary can delay the signals to produce an arbitrary $\hat{\theta}$ on the receiver, I can still provably bound $\hat{\theta}$ so as to notify the receiver of an insecure GIC. Under the reasonable assumption that an adversary cannot tamper directly with the relevant onboard oscillators, I can assume that the measurement of each oscillator linearly increases, as in Eq. (2.4) to derive Eq. (2.5) via measurement Eq. (2.1):

$$\begin{aligned} 0 &< \tau_4^l - \tau_3^l && (2.4)\\ 0 &< \tau_4^l - (t_3^l + \theta^l) \\ \theta^l &< \tau_4^l - t_3^l \ . && (2.5) \end{aligned}$$

And, as in Eq. (2.6) to derive Eq. (2.7) via measurement Eq. (2.1):

$$\begin{aligned} 0 &< \tau_2^l - \tau_1^l && (2.6)\\ 0 &< t_2^l + \theta^l - \tau_1^l \\ -(t_2^l - \tau_1^l) &< \theta^l \ . && (2.7) \end{aligned}$$

Combining Eqs. (2.5) and (2.7) yields bounds on $\theta^l$, as in Eq. (2.8): provably secure against man-in-the-middle attacks. If a man-in-the-middle adversary induces delays by increasing $t_2$ or $\tau_4$, the bounds become looser. $\tau_1^l$ is measured directly by the receiver and $t_3^l$ is protected with authentication cryptography in NTS; hence, both have integrity:

$$-(t_2^l - \tau_1^l) < \theta^l < \tau_4^l - t_3^l \; . \tag{2.8}$$

The spread between the bounds of $\theta^l$ in Eq. (2.8) is the round-trip time $\tau_4^l - t_3^l + t_2^l - \tau_1^l$, and $\hat{\theta}$ is the middle of that spread. Observing the round-trip time and specifying bounds on $\theta^l$ produce equivalent security on $\theta^l$ in the NTS context; however, the bounds of Eq. (2.8) are more useful in the GNSS TESLA context.

### Loose-Time Synchronization for Non-GNSS TESLA

In the GNSS context, $\Theta$ is immutably fixed constellation-wide, motivating the methods of this chapter to accommodate. A one-way signal cannot achieve secure time-synchronization [68]. Therefore, every TESLA-enabled GNSS receiver must utilize a GIC, and that GIC must be two-way synchronized without GNSS. If the receiver is disconnected, the GIC must have a low drift rate to enable security between the networked synchronizations during maintenance.

In the non-GNSS context, TESLA connection over a network can be two-way. The description of TESLA provides a simple protocol to bootstrap loose-time synchronization [80], which includes the first half of NTP (described in Algorithm 2.1 in Section 2.1.3). In a two-way TESLA-enabled system, the two individual communicating parties can set a safe hash-point-disclosure delay from Eq. (2.9), where $\Delta t$ is the length of the communication session and $B(\Delta t)$ is the maximum clock drift possible during the end of the communication session:

$$\Theta \; = \; t_2 \; - \; \tau_1 \; + \; B(\Delta t) \; . \tag{2.9}$$

This bootstrap is safe because if a man-in-the-middle increases the delay of the message (i.e., maliciously increases $t_2$), $\Theta$ increases to accommodate.

With the information transmitted with NTS, the parties achieve provable security by utilizing Eq. (2.9) to increase $\Theta$ until all parties certify that messages and commitments received after the release of the corresponding preimage hash point will be rejected. Since $\Theta$ is constellation-wide, authentication requires proof utilizing a GIC that the messages and commitments received after the release of the corresponding preimage hash point will be rejected.

### 2.1.4 Proving Receipt Safety under Adversary Model

The TESLA security argument derives from three steps. The message authenticity code is consistent with the message and delay-released hash point (***message integrity***). The delay released hash point is the correct preimage that derives a hash point signed via asymmetric cryptography (***message authenticity***). The message and message authenticity code arrived before the release of the corresponding hash point (***receipt safety***). To break TESLA security, the adversary must break either message integrity, message authenticity, or receipt safety. To break message integrity or message authenticity, the adversary must break the underlying cryptographic protocols (e.g., HMAC or SHA256), which is outside the scope of this work. This chapter focuses on the receipt safety part of TESLA.

Receipt safety means that it is safe to use the GIC to perform dispositive checks that determine whether to accept or reject a message to enforce that a message and its message authenticity code arrived before the release of the corresponding TESLA hash point. This work is necessary to ensure receipt safety because the protocol is broadcast-only and cannot adapt to user-specific hash point release delays. Instead, the user is responsible for maintaining a compliant GIC and using it correctly to accept or reject messages.

The adversary model has the following capabilities and limitations. The adversary has all the capabilities of a Dolev-Yao adversary, meaning they can overhear, intercept, and create messages but cannot break the underlying cryptographic primitives [37, 82]. Succinctly, the adversary carries the message. The adversary is perfectly synchronized to GNSS time. The adversary cannot tamper with the GIC. The clock output is strictly positive linear (except when adjusted during synchronization) but may have an offset. A non-negative, strictly increasing function bounds the maximum clock-offset growth over time during clock operation. When attempting to fool the receiver, the adversary may have zero latency but cannot break the message integrity or authentication of the underlying cryptographic primitives with an efficient algorithm. In an extreme case, this could mean that the adversary, receiver, and GNSS satellites are adjacent in orbit. And if a modern classical computer can perform a computation in polynomial time, the adversary can do that computation instantaneously.

For this chapter, I must disclaim protection for adversaries that can (1) manipulate the arrival time of ranging signals and (2) have those manipulated signals arrive at the receiver with a delay less than $\Theta$. These are the meaconing attacks from Section 1.2 who simply delay signals, and this is discussed further in Section 2.1.2. For a receiver to protect itself from these adversaries, its GIC certainty must be bound on the order of nanoseconds (1 nanosecond is about 3 meters of PNT manipulation). If the meaconer adversary needs to delay its manipulated signal by $\Theta$, then receiver GICs will provably detect their attack. It is a very difficult task to perform this attack on a receiver with a signal delay less than $\Theta$, meaning the methods and proofs of this chapter are still useful.

To fool a receiver, the adversary must induce a delay in the signal at least the length of the hash point delay time $\Theta$. After that delay, the adversary can instantaneously compute a forged message and transmit it to the receiver. The adversary wins and breaks receipt safety if it can fool a receiver into accepting a message after ***release*** of the delayed hash point. The challenge here is constructing a protocol that ensures receiver safety even when the GIC has an offset.

To prove that a protocol will ensure receipt safety, one must show that in the presence of the delay-capable adversary, the adversary still cannot fool the protocol into breaking receipt safety. The proof within this chapter is applicable for broadcast-only TESLA, whereas previous literature includes a generalized approach [68]. In this work, the mathematical arguments will follow a consistent structure. I introduce delays induced by the adversary at their election. These delays must be non-negative, or otherwise, the delays are not physically achievable. Sometimes, I may assume a condition on the offset of the GIC (or otherwise, it is assured by another safe procedure). Then, I show that the protocol requires the adversary to induce a negative delay to break receipt safety. Since negative delays are impossible, the protocol ensures receipt safety.

### 2.1.5 GIC Drift Model

Suppose that the GIC's clock offset over time is $\theta(t)$ and the clock offset at the last synchronization was $\theta^l$. For this chapter, there exists a non-negative, strictly increasing function $B(\cdot)$ that bounds the maximum GIC clock-offset growth over time during clock operation:

$$|\theta(t) - \theta^l| < B(t - t^l) \ . \tag{2.10}$$

Splitting Eq. (2.10) into the leading and lagging bounds and substituting Eq. (2.8) yields

$$\begin{aligned} -B(t - t^l) < \theta(t) - \theta^l < B(t - t^l) \ ; \\ -B(t - t^l) + \theta^l < \theta(t) < \theta^l + B(t - t^l) \ ; \\ -B(t - t^l) - (t_2^l - \tau_1^l) < \theta(t) < \tau_4^l - t_3^l + B(t - t^l) \ . \end{aligned} \tag{2.11}$$

These bounds that are a function of time will be abbreviated as

$$-B_l < \theta < B_u \ . \tag{2.12}$$

In Eq. (2.11), the uncertainty on $\theta$ bound by $B_l$ and $B_u$ will expand as time progresses since the last synchronization. For the proofs of this chapter, $B$, and therefore $B_l$ and $B_u$, are absolute bounds. The GIC's clock offset will never cross them, which is a non-physical assumption. I make this assumption for mathematical conciseness, but these proofs could be extended to accommodate a clock whose offset would not exceed the bounds with high probability, noting the corresponding decrease in security by that probability.

## 2.2 Selecting the Time Sync. Requirement

Section 2.1.1 discusses why this chapter is necessary, describing the relevant replay and delay attacks. Section 2.2.1 discusses how the selection of $\Theta$ implicates the overall design of the TESLA scheme. And Section 2.2.3 briefly overviews available clocks and their costs. From this section, a GNSS should select $\Theta$ almost independently of the design choices discussed in Chapters 3 and 4.

### 2.2.1 TESLA Design Implications

In between synchronizations, the receiver must ensure that its GIC meets a specific accuracy bound derived in Section 2.5. Based on the specific GIC characteristics, the receiver should form a model on the clock offset drift over time (see Section 2.1.5). When that model can no longer predict that the GIC meets the necessary accuracy bounds, the receiver must synchronize its GIC. Fig. 2.6 provides a conceptual diagram of this process.

In Fig. 2.6, the **blue** is a simulated random walk of a GIC, and the **cardinal** is a model on the GIC clock drift. For instance, the model could be the standard square-root-growing uncertainty bound on the expected random walk, as depicted in **cardinal** in Fig. 2.6. The model could be conservative (e.g., $5\sigma$ certain). But for this chapter, I assume there exists a model that can bound $|\theta|$ with 100% certainty. And when that model expands outside the safety boundary depicted in Fig. 2.6, the GIC will synchronize with a two-directional network.

As the GNSS designer elects to shrink $\Theta$, either the synchronization cadence will increase or the GIC hardware will become more expensive. At the same time, as the GNSS designer elects to shrink $\Theta$, the TTA will shrink as well. In addition, the selection of $\Theta$ is affected by the TESLA geometry, as conceptually depicted in Fig. 2.7.

In Fig. 2.7, Designs 1 and 2 differ by which messages are authenticated by which hash point. In Design 1, the hash point authenticates the second and third previous messages. In Design 2, the hash point authenticates the third and fourth previous messages. The $\Theta$ in Design 2 is larger than that of Design 1. Noting the behavior

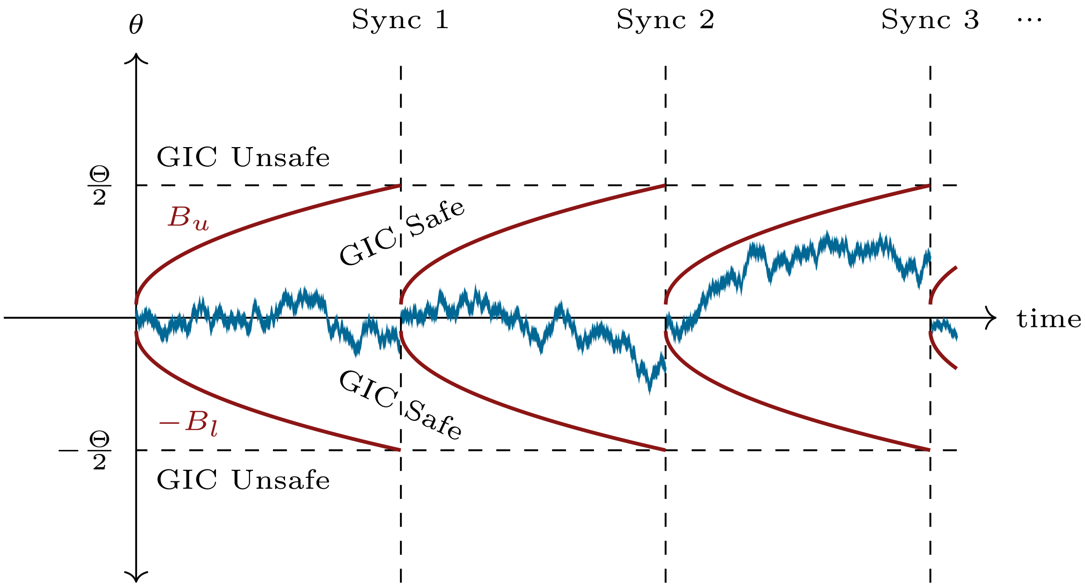


Figure 2.6: Conceptual Diagram Of GIC Offset And GIC Accuracy Bound Over Time.

In **blue** is the simulated clock offset of a GIC over multiple synchronization periods, which follows the typical random walk. In **cardinal** are known accuracy bounds on the GIC over time derived from a model of the GIC. Once the accuracy bounds on the GIC exceed a specific threshold based on the clock stability (see Section 2.2.3), the GIC must synchronize with a network (see Section 2.3). Provided the GIC always satisfies $|\theta| < \frac{\Theta}{2}$, the GIC is safe to use together with the procedures described in this chapter.

depicted in Fig. 2.6, because of Design 2's $\Theta$, it can either (1) accommodate receivers with less expensive GIC or (2) worse access to network synchronization.

For the most part, the selection of $\Theta$ can be done independently of the geometry, allowing the designer to focus on accommodating synchronization cadence and clock expense. Fig. 2.7 depicts how $\Theta$ is not independent of the geometry; rather, there are limitations resulting from the cadence and order of messages. However, $\Theta$ can be increased as much as needed to accommodate the synchronization cadence and clock expense, and as the data rate increases, the effect of message cadence and order diminishes (which is why I said nearly independent). So, for the most part, the GNSS designer must select $\Theta$ to balance the expense of receiver GIC, GIC access to

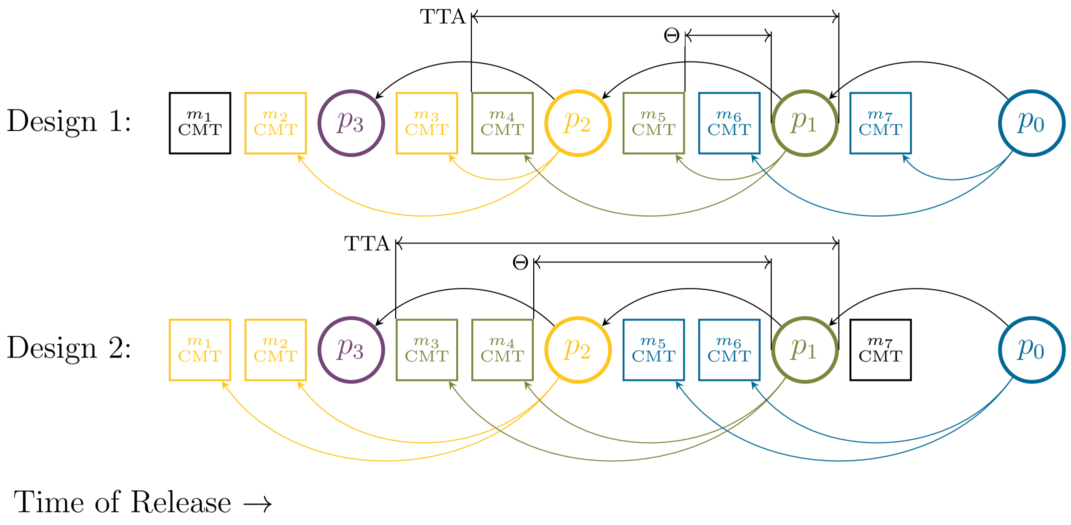


Figure 2.7: Conceptual Design Comparing Two TESLA Designs With Differing Θ.

The difference between Design 1 and Design 2 is the correspondence relation between messages and CMTs and hash points. Because Design 2 hash points authenticate messages earlier than Design 1, Design 2's Θ is larger, accommodating users with a worse GICs or worse network synchronization access. Because Design 1 hash points authenticate messages later than Design 2, Design 1's Θ is smaller, enabling a faster time to authentication.

a network to synchronize, and the TTA afforded to receivers.

### 2.2.2 Message or CMT First?

So far, the conceptual diagrams have depicted schemes for a message, commitment, and then hash point release cadence. Θ has been the smallest time distance from any CMT's last bit to the first bit of the associated hash point. And each message authentication code (MAC) has been explicitly labeled a commitment-MAC tag (CMT). Provided that it is a CMT (and not just a MAC), the GNSS designer may elect to have the commitments come before the messages. When the receiver must have the message, CMT, and hash point before using the message, changing the message and CMT order does not affect TTA. This section considers whether swapping their order can increase the Θ. In the context of shorter CMTs for GNSS, swapping the order does ***not*** provide an advantage for Θ.

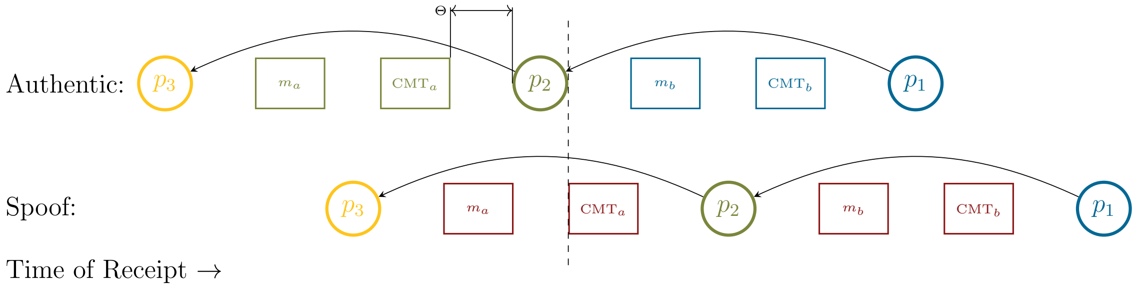


Figure 2.8: Conceptual Diagram Of Spoof Of Message-CMT Cadence.

The adversary delays the signal to the receiver by greater than $\Theta$ to the point that $\text{CMT}_a$ will arrive after the adversary has access to $p_2$. The adversary submits an arbitrary message $m_a$ to the receiver. After the adversary receives $p_2$, the adversary forges $\text{CMT}_a = \text{CMF}(p_2, m_a)$.

Fig. 2.8 depcits a spoofing attack on the message-commitment-hash-point cadence. As discussed in Section 2.1.1, the adversary can start undermining the security once the delay exceeds $\Theta$. For intuitive convenience, Fig. 2.8 depicts the point where the CMT arrives after the hash point. The adversary has access to $p_2$ to submit a forged $\text{CMT}_a$ to the receiver.

Now, in Fig. 2.9, the message and commitment cadence order are swapped. This time, the adversary must first commit to a forged $\text{CMT}_a$. Then, once the adversary knows $p_2$, it must solve for $m_a$

$$\text{CMT}_a = \text{CMF}(p_2, m_a)\ . \tag{2.13}$$

The forged $\text{CMT}_a$ submitted to the receiver in Fig. 2.9 can be random. It does not matter because the adversary can then attempt to solve Eq. (2.13) via rapid guessing (including in parallel). For a $b$-bit CMT, the adversary should be able to solve Eq. (2.13) with $b$ trials, in expectation. In the GNSS context, systems are selecting CMT sizes on the order of 32 bits [13, 39], and 32-bit searches can be completed with modern hardware in seconds. Therefore, it is not secure to increase $\Theta$ when swapping the order of messages and commitments (unless the CMT length is on the order of 128 bits). The later sections of this chapter will set $\Theta$ to be based

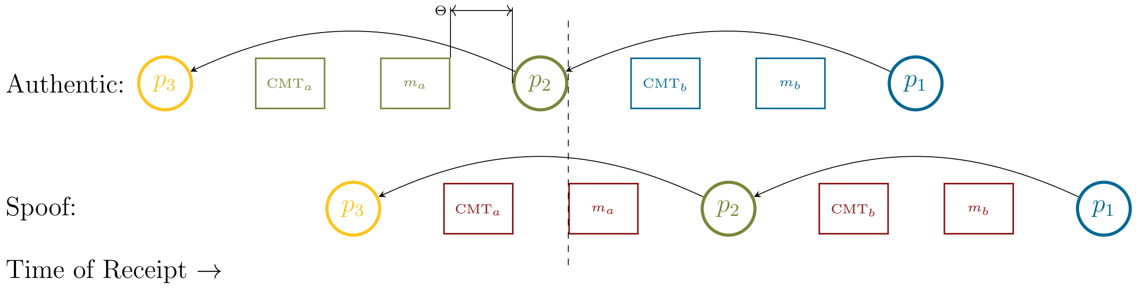


Figure 2.9: Conceptual Diagram Of Spoof Of CMT-Message Cadence.

The adversary delays the signal to the receiver by greater than $\Theta$ to the point that $m_a$ will arrive after the adversary has access to $p_2$. The adversary selects random bits to form a random CMT$_a$ and submits it to the receiver. After the adversary receives $p_2$, the adversary solves Eq. (2.13) for $m_a$. The adversary can use its parallel computational resources to solve this equation in the midst of the spoofing delay. Upon finding a succesful $m_a$, the adversary can submit a forged $m_a$.

on the latest distributed object among all messages and CMTs.

### 2.2.3 GIC Types

In selecting $\Theta$, the GNSS must consider the concerns of any disconnected receivers [17]. If the receivers utilizing the GNSS TESLA features have access to a network, then it will be trivial for the receivers to synchronize the GIC to the order of milliseconds. These network receivers can utilize cheap oscillators that already pervade microelectronics. The tricky part is for receivers that must maintain accuracy without easy access to a network connection.

For disconnected receivers, the smaller the $\Theta$, the more arduous the design concern. And the longer the time between GIC maintenance sessions, the more arduous the design concern. From Section 2.4, the clock offset must maintain $-\frac{\Theta}{2} < \theta < \frac{\Theta}{2}$ between maintenance sessions. Hence, the larger the $\Theta$ and the faster the GIC maintenance cadence, the easier this requirement is to satisfy.

Disconnected receivers will have hardware concerns. These can include the cost of the receiver, the space in which a receiver must fit, the required heat dissipation for the receiver, and the power consumption. As cost increases, the space, heat generated,

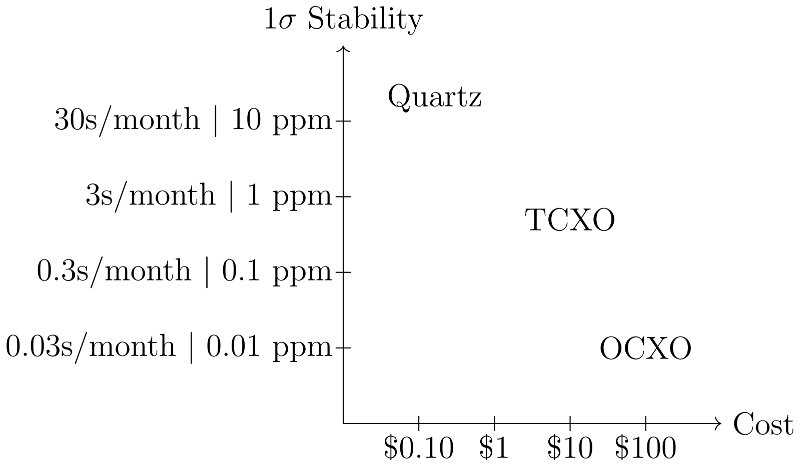


Figure 2.10: Conceptual Graph Depicting The Cost And Stability Trends Of Various Clock Types.

As cost increases, clock stability gets better. In selecting a $\Theta$, the GNSS designer must balance TTA with the inconvenience caused to the receiver in maintaining a better GIC. These inconveniences can include the cost, space, heat dissipation, and power consumption, among others.

and power consumption also go up. Fig. 2.10 provides a conceptual graph of common clock types' general cost and accuracy trends.

As an example, with the Satellite-based Augmentation System (SBAS) TESLA proposal [13], $\Theta = 6$. Hence, the receivers must maintain a GIC so that $-3 < \theta < 3$ in between maintenance sessions. It has been suggested that aviation SBAS receivers perform maintenance every 56 days to align with the International Civil Aviation Organization (ICAO) Aeronautical Information Regulation and Control (AIRAC) schedule [78]. Adding to the complexity of this requirement is how aviation frequently undergoes wide temperature variation. From Fig. 2.10, this would require that the SBAS receiver GIC be a TCXO or an OCXO.

## 2.3 GIC Procedures

For the implementor's convenience, I aggregate all the concepts described in this chapter into this section. Section 2.3.1 contains Algorithm 2.2 to synchronize GICs securely. Section 2.3.2 contains Algorithm 2.3 to utilize the GIC correctly. Section 2.4 contains the derivation and proof of security. Section 2.5 contains the derivation and proof of security. Algorithms 2.2 and 2.3 contain the two algorithms required from a timing perspective.

### 2.3.1 GIC Synchronization

Algorithm 2.2 contains the following features. First, a modified NTS query that accounts for the vulnerabilities resulting from disclosing $\tau_1$ (see Section 2.6). Second, the clock drift correction is bounded according to Section 2.5.2. Third, the next synchronization time is computed to always satisfy safety conditions between synchronizations according to Section 2.5.1. I refer to Section 2.6 for a suggestion on how to modify this procedure if it is not acceptable to shut down after a denial of synchronization service.

### 2.3.2 GIC Use on Message Stream

Algorithm 2.3 contains the procedures to ensure the receipt safety of the messages within broadcast-only TESLA correctly. If a message has receipt safety, the receiver must also perform the additional TESLA checks to ensure message integrity and authenticity.

Algorithm 2.3 is generalized to accomidate any message, commitment, and hash point cadence. If the GNSS design has a simple and rigid cadence (e.g., a single message, then commitment, then hash point), then the max statement in Algorithm 2.3 is not needed.

**Algorithm 2.2:** GIC Synchronize.

```
GNSS provider and receiver establish an asymmetrically encrypted and authenticated channel for NTS synchronization $l$.
Receiver draws a nonce $\eta^l$ to associate the return message and deter replay and denial of service.
Receiver sends message $m_1^l = (\eta^l, s^{\text{receiver}_1,l})$ specifically omitting $\tau_1$ or replacing the field with any value independent of $\tau_1^l$, and $s_1^{\text{receiver},l}$ as the receiver's authentication signature on $(\eta^l, )$.
Receiver measures the moment of sending message $m_1^l$ as $\tau_1^l$ and holds $\tau_1^l$ in strict confidence.
Provider records $t_2^l$, the time of receipt of the message $m_1^l$.
Provider sends message $m_2^l = (\eta^l, t_2^l, t_3^l, s_2^{\text{provider},l})$ back to receiver, where $s_2^{\text{provider},l}$ is provider's authentication signature on $(k, t_2^l, t_3^l)$.
Receiver records $\tau_4^l$ at the moment of receipt of message $m_2^l$.
if $\Theta < \tau_4^l - t_3^l + t_2^l - \tau_1^l$ from Eq. (2.46) then
    Receiver cannot authenticate any future messages and has leaked information about its clock offset (noting the alternative presented in Section 2.6 and its limitations on provable safety).
    return
end
The receiver adjusts its GIC by subtracting any $\delta\theta^l$ from its current GIC output that satisfies Eq. (2.44):
    $$\tau_4^l - t_3^l - \frac{\Theta}{2} < \delta\theta^l < -(t_2^l - \tau_1^l) + \frac{\Theta}{2}.$$
I suggest the of midpoint of the bounds in Eq. (2.44): $\delta\theta = \frac{1}{2} \cdot (\tau_4^l - t_3^l - t_2^l + \tau_1^l)$.
The receiver computes the latest acceptable next synchronization time $t$ by solving the following program using constraint Eqs. (2.34) and (2.37) adjusted by the $\delta\theta^l$ correction of the previous step.
    $$\begin{aligned} t^{l+1} = \quad & \max \quad t \\ \text{subject to} \quad & \tau_4^l - \delta\theta^l - t_3^l + B(t - t^l) < \frac{\Theta}{2} \\ & -\frac{\Theta}{2} < -(t_2^l - \tau_1^l + \delta\theta^l) - B(t - t^l) \end{aligned}$$
```

**Algorithm 2.3:** GIC Receipt Safety Check.

```
Receiver presumes its GIC satisfies the following condition from Eq. (2.23)
  via previous successful execution of Algorithm 2.2 and with the GIC's
  stability thereafter:
      $-\frac{\Theta}{2} < -B_l < \theta < B_u < \frac{\Theta}{2}.$
Receiver uses the GNSS system design to correctly associate messages,
  commitments (e.g., CMTs, watermarks), and delay-released hash points as
  (message $m$, commitment $h$, hash point $p$) Tuples.
forall (message m, commitment h, hash point $p$) tuples do
    Measure the receipt time of commitment h as $\tau_{\mathrm{h}}$ and message m as $\tau_{\mathrm{m}}$
      with the GIC.
    Recall the correct release time of the corresponding hash point $p$ as $t_{\mathrm{p}}$.
    if $\max(\tau_{\mathrm{h}}, \tau_{\mathrm{m}}) < t_{\mathrm{p}} - B_l$ then
        m has receipt safety. Perform the additional TESLA security checks
          (e.g., h = H(p, m)) to determine message integrity and message
          authenticity.
    else
        m does not have receipt safety, so m does not have authenticity.
    end
end
```

## 2.4 Safe Use of a GIC

Due to the broadcast-only nature of GNSS, receivers must use a GIC to enforce that all authenticated information is received before distribution of the associated hash point. Based on the TESLA scheme's authenticated metadata and GIC, a receiver should be able to associate messages, commitments, and hash points together [13]. Fig. 2.11 depicts the message-commitment-hash point cadence between the provider and receiver. As the adversary achieves zero latency, $\varepsilon \to 0$ and Fig. 2.11 would depict vertical transmission arrows for the case of instantaneous message transmission, representing the most conservative case in the future bound derivations. Moreover, $\Theta = \min(t_p - t_h, t_p - t_m)$ is a constant for all tuples of a message, commitment, and hash point.

Safe use of the GIC requires that it be GNSS-independent. It should be modified

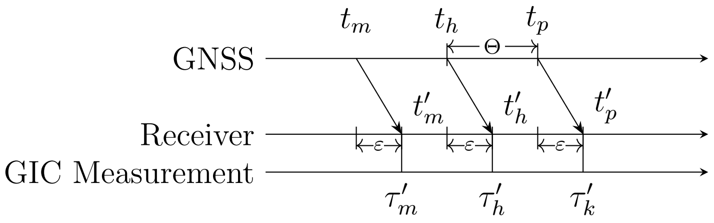


Figure 2.11: Conceptual Diagram Of The Message, Commitment, And Hash Point Release Cadence.

The diagram depicts a TESLA message, CMT, delay-release hash point transmission cadence from provider to receiver ($m$, $h$, $p$ in the diagram, respectively). The transmission arrows approach vertical for the conservative case with a zero-latency adversary as $\varepsilon \to 0$, which aids in the conciseness of the proofs provided in Section 2.4. The fixed commitment delay is set constellation-wide as $\Theta = \min(t_p - t_h, t_p - t_m)$ among all $(m, h, p)$tuples, determining the requirements on receiver onboard, GNSS-independent clocks.

using the procedure described in Section 2.5.2 and never changed with broadcast-only GNSS. To prohibit the acceptance of forgeries, the receiver must ensure two conditions (noting Section 2.4.2), which are incorporated in Algorithm 2.3. First, the GIC must synchronized such that

$$-\frac{\Theta}{2} < -B_l < \theta \ . \tag{2.14}$$

Second, the GIC must assert that each message, commitment, and hash point tuple satisfies

$$\max(\tau'_m, \tau'_h) < t_p - B_l \ . \tag{2.15}$$

The conditions of Eqs. (2.14) and (2.15) are sufficient to have receipt safety.

Eq. (2.15) is intended to be a broad condition that accommodates many TESLA-enabled GNSS system designs. For instance, $\max(\tau'_m, \tau'_h)$ is the latest release time of any piece of information corresponding to hash point $p$ when there are multiple messages and commitments associated with a hash point. If the system has a simple cadence like Fig. 2.11, then the condition simplifies.

I note that in Eq. (2.15), $\tau'_m$ and $\tau'_h$ are measurements with the GIC that meet

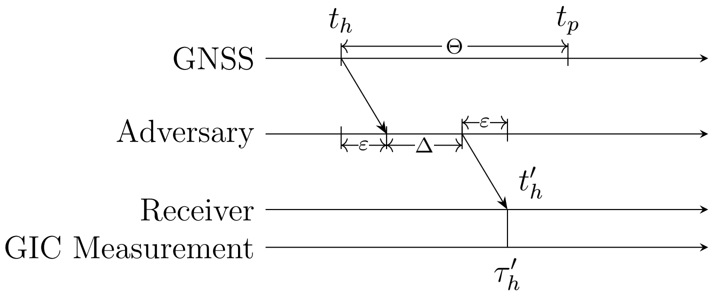


Figure 2.12: Conceptual Diagram Of Delay Spoof.

The diagram depicts the spoofing scenario as a GNSS message traverses to the receiver. GNSS transmits commitment $h$, which the adversary delivers to the receiver with an adversary-selected delay $\Delta$. The adversary and scenario have latencies $\varepsilon$. I allow $\varepsilon \to 0$ for the sake of proving the worst case. Commitment $h$ is transmitted and received at time $t_h$ and $t'_h$, respectively, in the clock frame provided by GNSS. Commitment $h$ is measured by the GIC to have been received at $\tau'_h$ under a clock offset of $\theta$. If the receiver accepts $h$ after GNSS releases $p$, then the receiver would accept an arbitrary forgery, which occurs when $\Delta \geq \Theta$.

the condition of Eq. (2.14). Whereas $t_p$ is determined by the schedule that the GNSS Provider adheres to.

After determining a message has receipt safety, the receiver must perform additional cryptographic TESLA checks to ensure message integrity and authentication.

### 2.4.1 Proof of Receipt Safety

In this section, I show that Eqs. (2.14) and (2.15) are sufficient to have receipt safety. When the message and commitment are received before the release of the hash point, the message and commitment are sufficient for receipt safety.

As conceptually diagrammed in Fig. 2.12, GNSS releases the commitment $h$ at $t_h$, where the adversary observes, modifies, and then delivers it to the receiver. The receiver receives commitment $h$ at $t'_h$, which is measured by the receiver GIC as time $\tau'_h$. Like Fig. 2.11, Fig. 2.12 assumes that the commitment $h$ is the latest released object associated with $p$ without loss of generality.

I let the adversary have zero latency, so each $\varepsilon \to 0$, and the adversary is perfectly synchronized with GNSS. In Fig. 2.12, the adversary wins if the receiver accepts a commitment $h$ after GNSS releases the corresponding hash point $p$. This corresponds to $\Delta \geq \Theta$ when the adversary induces sufficient delay to forge an arbitrary message and commitment with knowledge of the released $p$.

I will prove that if a receiver certifies that its GIC synchronization meets the condition of Eq. (2.14) and enforces the condition of Eq. (2.15), then the message and its commitment arrived before the release of the hash point (i.e., $\max(t'_m, t'_h) < t_p$). Therefore, the message has a receipt safety.

I start with the enforced condition

$$\max(\tau'_m, \tau'_h) < t_p - B_l \ . \tag{2.15}$$

Then, by substituting the measurement equation, I have

$$\max(t'_m, t'_h) + \theta < t_p - B_l \tag{2.16}$$

$$\theta < t_p - B_l - \max(t'_m, t'_h) \ . \tag{2.17}$$

Next, I substitute the presumed clock synchronization condition and simplify it with

$$-B_l < t_p - B_l - \max(t'_m, t'_h) \tag{2.18}$$

$$\max(t'_m, t'_h) < t_p \ . \tag{2.19}$$

Since the message and commitments were received ***in the GNSS clock frame*** before the release of the corresponding hash point, it has receipt safety under TESLA.

### 2.4.2 Alternative Condition Design

From Eq. (2.14), the receipt safety conditions relate to $\frac{\Theta}{2}$. But what about beyond check $\max(\tau'_m, \tau'_h) < t_p - B_l$ and synchronization condition $\theta > -\frac{\Theta}{2}$? And why $\frac{\Theta}{2}$, as opposed to $\Theta$?

From an intuition point of view, the $\frac{\Theta}{2}$ comes from the following scenario. Suppose a receiver's clock is $\frac{\Theta}{2}$ behind. If an adversary delays the messages by $\Theta$, it (1) can forge messages, and (2) the message will induce the receiver to believe its clock is $\frac{\Theta}{2}$ ***ahead***. $\frac{\Theta}{2}$ ahead is also within the safe condition. Hence, the entire safe region cannot be larger than $\Theta$, and $\frac{\Theta}{2}$ spreads the uncertainty in both directions evenly for unbiased clock drift.

Without loss of generality, let's suppose that the GNSS system has a single-object cadence: $m$, $h$, and $p$. Therefore, the decision boundary becomes $\tau'_h = t_p - B_l$. There are two independent states in this scenario: $\Delta$ and $\theta$. The adversary may elect any $\Delta \geq 0$, and the clock drifts $\theta$ may be any value, subject to the clock offset bounds. There is no receipt safety when $\Delta \geq \Theta$, where an adversary may forge an arbitrary message and commitment with the released hash point. Substituting decision boundary $\tau'_h = t_p - B_l$ yields functions of the state variables with

$$\begin{aligned}
\tau'_h &= t_p - B_l & (2.20)\\
t'_h + \theta &= t_p - B_l\\
t_h + \Delta + \theta &= t_p - B_l & (2.21)\\
\Delta + \theta &= t_p - t_h - B_l\\
\Delta + \theta &= \Theta - B_l\ . & (2.22)
\end{aligned}$$

From states $\Delta$ and $\theta$, I plot them with Fig. 2.13 to visualize a receiver's operating condition to assist with design. The check must ensure that no adversary-selected $\Delta$ exists under any possible clock condition $\theta$ that would have the receiver accept a forgery. Because $\theta$ and $\Delta$ contribute equally to $\tau'_h$ in Eq. (2.21), the slope of the line in Fig. 2.13 cannot be changed. However, one could raise or lower the decision line, provided the synchronization requirement is changed.

For instance, a receiver could enforce the condition $\tau'_h < t_p$ and require that $\theta > -\Theta$. This corresponds to moving each boundary line and shape in the figure to the left by $\frac{\Theta}{2}$. The issue with doing this change is the potential for false alarms. If I reasonably assume that, at synchronization, $\theta = 0$, then depending on the clock drift, the check would have a 50% chance of false alarm. Optimizing this condition may be useful if the receiver knows its clock drift rate is biased, but otherwise, it would be best to use the original condition and proof above.

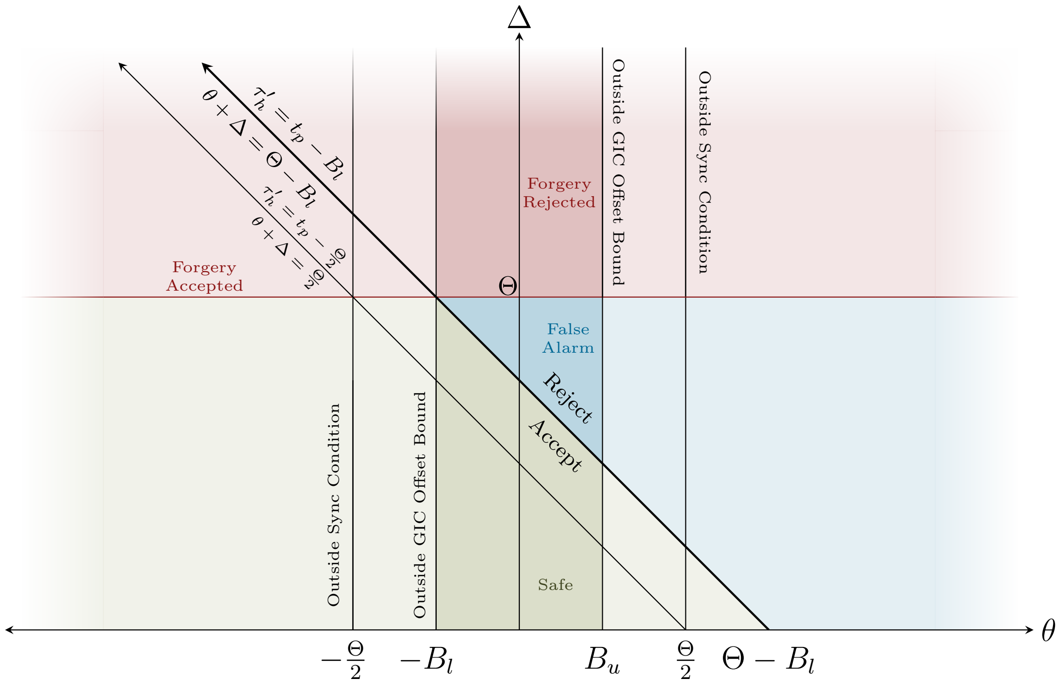


Figure 2.13: Conceptual Diagram Depicting Safety Regions Among Adversary-selected Delay And GIC Clock State.

Among the the adversary-selected state $\Delta$ and the GIC drift state $\theta$, there must be constraints to prohibit forgery. The figure shows the safety condition line and labels regions of interest. While my selected decision line is shown, the GNSS designer can shift the regions left or right. Here, the uncertainty of the clock drift is evenly spread to accommodate unbiased clock drift in the context of false alarms. The entire region Forgery Accepted is outside the time synchronization bound. As time progresses, the decision the decision line will move down, expanding the False Alarm region until the GIC must synchronize.

Choosing a decision boundary with $\frac{\Theta}{2}$ spreads the slack evenly. For the rest of this work, I adopt this evenly split condition. This also means that Eq. (2.14) must be amended to bind the leading case to accommodate false alarms to become

$$-\frac{\Theta}{2} < -B_l < \theta < B_u < \frac{\Theta}{2} \ . \tag{2.23}$$

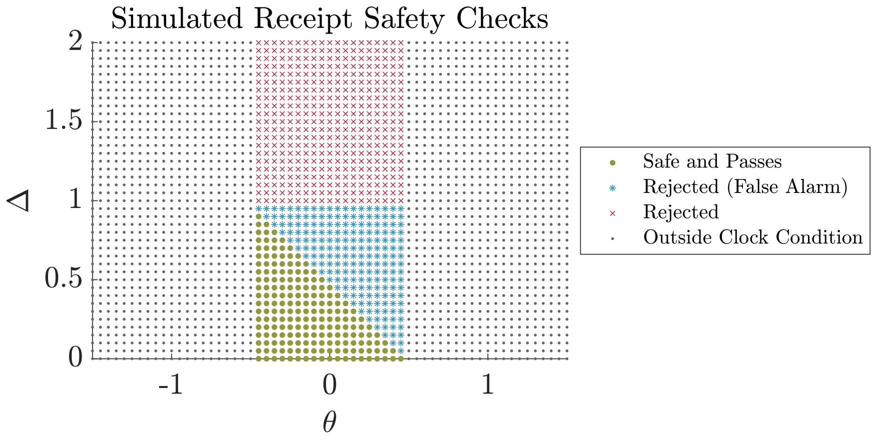


Figure 2.14: Results From Simulations Using A Representative Set Of $\theta$ And $\Delta$ To Validate The Receipt Safety Check Of Eq. (2.15).

### 2.4.3 Experimental Validation

In this section, I validate the checks proposed in Sections 2.4 and 2.5 by simulation. To simulate, I generate a representative set of scenarios among $\theta$ and $\Delta$. In the simulations, I set $\Theta = 1$, meaning the forgery condition is $\Delta \geq 1$. I simulate clocks with offsets between -2 and 2. In addition to the adversary-induced delays, I add a 0.01 latency within every simulated transmission.

In Fig. 2.14, for each $\theta$ and $\Delta$, I evaluate the check of Eq. (2.15). I observe no condition leading to the acceptance of a forged message within the bounds of an acceptable clock synchronization.

### 2.4.4 Addressing Multi-cadence TESLA Time Synchronization

There are GNSS authentication proposals that will include multiple, independent authentication schemes [15]. The schemes will operate at different TTA and $\Theta$ to accommodate different receiver use cases. For instance, there are two authentication TESLA instances for users with and without an internet connection [1]. These two use cases determine the appropriate precision for the GIC: a receiver without an internet connection might need a low-drift clock. Or, a signal scheme could accommodate standard drift rates and lower frequency synchronizations by increasing the $\Theta$. A clock becomes vulnerable after its drift accumulates into lagging beyond provider time by $\frac{\Theta}{2}$; therefore, increasing $\Theta$ decreases that likelihood and increases the time between maintenance.

This section addresses whether a slower TESLA instance can assist or bootstrap a faster TESLA instance. ***It cannot***. Suppose that the faster TESLA instance has $\Theta_{\mathbf{blue}}$ and the slower TESLA instance has $\Theta_{\mathbf{green}}$ (i.e., $\Theta_{\mathbf{blue}} < \Theta_{\mathbf{green}}$). I will show that a receiver whose clock is rated for the slower TESLA can never assert the security of the faster TESLA instance. A slow TESLA scheme could redundantly sign the fast TESLA instance to save bandwidth for slow-TESLA receivers [8]. In that case, there is no time-to-authentication advantage to the slow-TESLA receiver. In other words, meeting the loose-time synchronization requirement for slow TESLA and successfully authenticating a slow-TESLA message does not aid in satisfying the fast-TESLA loose-time synchronization requirement to utilize faster authentication. A receiver rated for a slow-TESLA instance is forever required to wait for the slow-TESLA delay release time, asserting receipt safety.

To demonstrate the no-time-to-authenticate-advantage property of multiple TESLA instances, consider a signal with two TESLA instances redundantly authenticating the same set of messages. One of the instances has a larger $\Theta$ to accommodate receivers with lower-quality GICs. Fig. 2.15 provides a conceptual diagram of the signal and two spoofs of different adversary-induced delays. Table 2.1 provides the GIC offset conditions where forgery would be successful.

In Fig. 2.15, the signal contains two TESLA instances: **green** and **blue**. Each message is signed by both TESLA instances. The hash point release for the **blue** TESLA instance comes one m-h-p tuple later, and the hash point release for the **green** TESLA instance comes two m-h-p tuples later. In Fig. 2.15, $\Theta_{\mathbf{blue}} < \Theta_{\mathbf{green}}$. Message $m_1$ has a **blue** CMT derived from $p_2$ and a **green** CMT derived from $p_1$.

In Table 2.1, Scenario 2 is the GIC condition that demonstrates it is not secure to believe that a successful slow-TESLA authentication allows a slow-TESLA receiver to authenticate at the cadence afforded to fast-TESLA receivers. A GIC in a safe condition for a slow-TESLA instance but in a broken condition for the fast-TESLA

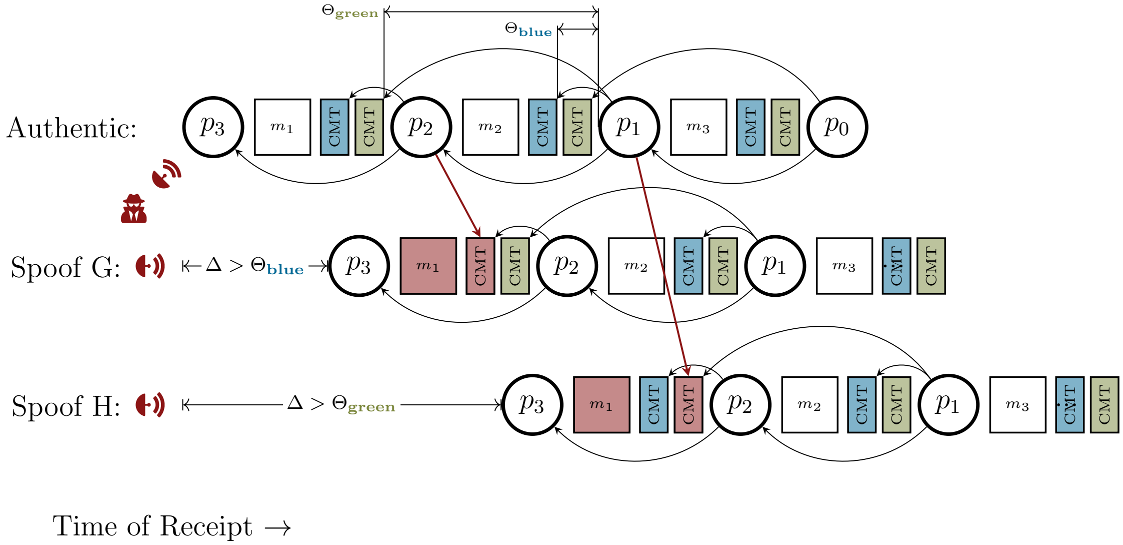


Figure 2.15: Conceptual Diagram Of Attack On Multi-cadence TESLA.

The diagram depicts a signal where messages are redundantly authenticated by two TESLA instances (**blue** and **green**) where one of the instances has a larger $\Theta$ to accommodate receivers with worse GICs. The diagram shows three cases: authentic, Spoof G, and Spoof H. The adversary observes the authentic and can replay the signal with spoofed messages. Spoof G's delay is such that $\Theta_{\textbf{blue}} < \Delta < \Theta_{\textbf{green}}$. Spoof H's delay is such that $\Theta_{\textbf{green}} < \Delta$. Table 2.1 describes the GIC conditions where spoofed messages would not be detected by one or both TESLA instances.

instance would accept a fast-TESLA forgery. A slow-TESLA receiver checking the fast-commitments ***and*** the slow-commitments would detect the forgery in Scenario 2 at the slow-TESLA pace. Hence, there is no TTA advantage, and a receiver should withhold its authenticated flag until then. If the slow-TESLA receiver in Scenario 2 does not check the slow-TESLA commitments, then it would accept forgeries. I provide proof of the existence of such an attack in the next section.

### Proof of Potential Forgery with Multicadence TESLA

Fig. 2.16 provides a conceptual diagram of the attack scenario. Suppose a signal has two TESLA instances, $\Theta_{\textbf{green}}$ and $\Theta_{\textbf{blue}}$, and $\Theta_{\textbf{blue}} < \Theta_{\textbf{green}}$. To simplify the proof, without loss of generality, I assume that they share the same delayed hash points and

| Scenario | GIC Condition | Description |
|---|---|---|
| 1 | $-\frac{\Theta_{\textbf{green}}}{2} < -\frac{\Theta_{\textbf{blue}}}{2} < \theta$ | Spoofed messages are detected (up to the security level) for both the **blue** and **green** TESLA instances. |
| 2 | $-\frac{\Theta_{\textbf{green}}}{2} < \theta < -\frac{\Theta_{\textbf{blue}}}{2}$ | Spoofed messages are detected (up to the security level) with the **green** TESLA instance. If the adversary delays the signal greater than $\Theta_{\textbf{blue}}$ and the receiver's GIC offset is such that $-\frac{\Theta_{\textbf{green}}}{2} < \theta < -\frac{\Theta_{\textbf{blue}}}{2}$, the adversary can forge messages that pass the **blue** but not the **green** TESLA instances. In Spoof G of Fig. 2.15, the adversary listens to $p_2$ to forge the **blue** CMT for a delayed $m_1$. |
| 3 | $\theta < -\frac{\Theta_{\textbf{green}}}{2} < -\frac{\Theta_{\textbf{blue}}}{2}$ | If the adversary delays the signal greater than $\Theta_{\textbf{green}}$ and the receiver's GIC offset is such that $\theta < -\frac{\Theta_{\textbf{green}}}{2} < -\frac{\Theta_{\textbf{blue}}}{2}$, the adversary can forge messages that pass both TESLA instances. In Spoof G of Fig. 2.15, the adversary listens to $p_1$ to forge the **blue** CMT for a delayed $m_1$. |

Table 2.1: GIC Conditions For Which The Spoofs Of Fig. 2.15 Would Be Successful.

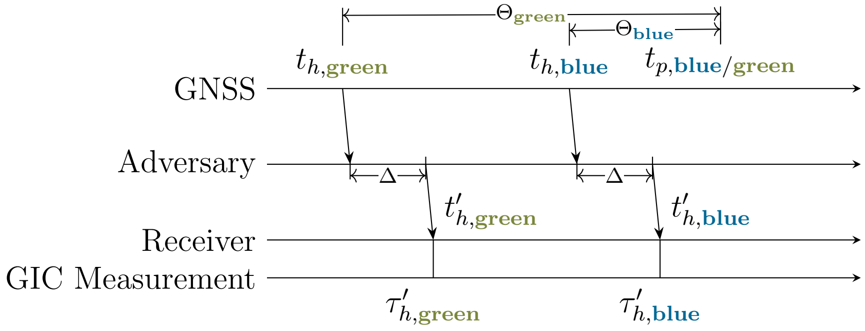


Figure 2.16: Conceptual Diagram Of An Attack Against Multi-cadence TESLA.

A conceptual diagram of the attack against a multi-cadence TESLA receiver that does not meet the synchronization condition of the tighter TESLA instance. Latencies are not depicted, but they approach 0.

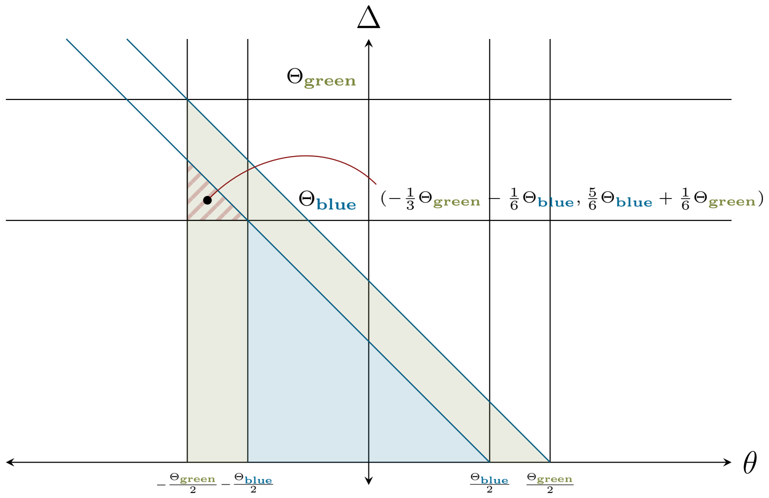

$\Delta$
$\Theta_{\mathbf{green}}$
$\Theta_{\mathbf{blue}}$
$(-\frac{1}{3}\Theta_{\mathbf{green}} - \frac{1}{6}\Theta_{\mathbf{blue}}, \frac{5}{6}\Theta_{\mathbf{blue}} + \frac{1}{6}\Theta_{\mathbf{green}})$
$\theta$
$-\frac{\Theta_{\mathbf{green}}}{2}$
$-\frac{\Theta_{\mathbf{blue}}}{2}$
$\frac{\Theta_{\mathbf{blue}}}{2}$
$\frac{\Theta_{\mathbf{green}}}{2}$

Figure 2.17: Conceptual Diagram Depicting The Insecure State Regions For Incorrect Multi-cadence TESLA.

> The diagram depicts the unsafe region when a clock certified for one TESLA instance safety region attempts to enforce checks on a tighter TESLA instance. The diagram shows regions among the adversary-selected state $\Delta$ and the onboard, GNSS-independent clock drift state $\theta$. The figure depicts the safety condition lines, and the hashed triangle is the unsafe region where a forgery can occur. The centroid of the region is marked as an example state that will induce a forgery.

have a message-commitment-commitment-hash point cadence. I examine the scenario where a receiver uses a successful authentication via the **green** instance to certify the receipt safety of the **blue** TESLA information. To prove no receipt safety, I will show that there exists an adversary induced $\Delta \geq \Theta_{\mathbf{blue}}$ and clock state $\theta$ that is safe for the **green** instance Eq. (2.24), passes the check for the **green** instance Eq. (2.25), passes the check for the **blue** instance Eq. (2.26), but breaks the receipt safety condition for the **blue** instance Eq. (2.27). In order, this corresponds to

$$
\begin{align}
-\frac{\Theta_{\mathbf{green}}}{2} &< \theta \tag{2.24}\\
\tau'_{h,\mathbf{green}} &< t_p - \frac{\Theta_{\mathbf{green}}}{2} \tag{2.25}\\
\tau'_{h,\mathbf{blue}} &< t_p - \frac{\Theta_{\mathbf{blue}}}{2} \tag{2.26}\\
t'_{h,\mathbf{blue}} &\geq t_p \ . \tag{2.27}
\end{align}
$$

From Eq. (2.26), I substitute the Eq. (2.1),

$$
\begin{align}
\tau'_{h,\mathbf{blue}} &< t_p - \frac{\Theta_{\mathbf{blue}}}{2} \tag{2.26}\\
t'_{h,\mathbf{blue}} + \theta &< t_p - \frac{\Theta_{\mathbf{blue}}}{2} \notag\\
t_{h,\mathbf{blue}} + \Delta + \theta &< t_p - \frac{\Theta_{\mathbf{blue}}}{2} \notag\\
\Delta + \theta &< t_p - t_{h,\mathbf{blue}} - \frac{\Theta_{\mathbf{blue}}}{2} \notag\\
\Delta + \theta &< \frac{\Theta_{\mathbf{blue}}}{2} \ . \tag{2.28}
\end{align}
$$

Starting with Eq. (2.27), I substitute and simplify,

$$
\begin{align}
t'_{h,\mathbf{blue}} &\geq t_p \tag{2.27}\\
t_{h,\mathbf{blue}} + \Delta &\geq t_p \notag\\
\Delta &\geq t_p - t_{h,\mathbf{blue}} \notag\\
\Theta_{\mathbf{blue}} &\leq \Delta \ . \tag{2.29}
\end{align}
$$

Combining Eqs. (2.28) and (2.29) yields

$$\Delta + \theta < \frac{\Theta_{\textbf{blue}}}{2} \tag{2.28}$$

$$\Delta < \frac{\Theta_{\textbf{blue}}}{2} - \theta$$

$$\Theta_{\textbf{blue}} < \frac{\Theta_{\textbf{blue}}}{2} - \theta$$

$$\theta < -\frac{\Theta_{\textbf{blue}}}{2} \; . \tag{2.30}$$

Combining Eqs. (2.24) and (2.28) to (2.30), I have the necessary $\Delta$ and $\theta$ to induce a forgery:

$$\Delta + \theta < \frac{\Theta_{\textbf{blue}}}{2} \tag{2.28}$$

$$\Theta_{\textbf{blue}} \leq \Delta \tag{2.29}$$

$$-\frac{\Theta_{\textbf{green}}}{2} < \theta < -\frac{\Theta_{\textbf{blue}}}{2} \; . \tag{2.31}$$

When generating a plot like Fig. 2.13, I observe the triangle of Fig. 2.17. One can verify that each of the points in the interior (and some of the boundaries) satisfy Eq. (2.24) through Eq. (2.27). The figure includes the triangle centroid, which will always satisfy the conditions for forgery for the tighter TESLA instance.

## 2.5 Safe Synchronization for a GIC

In this section, I provide two useful and dispositive procedures for broadcast-only TESLA loose-time synchronization for TESLA-enabled GNSS, and I prove their security under the adversary. In Section 2.5.1, I describe how to certify the safety of the GIC performing the receipt safety checks without making any adjustments to the clock. A receiver must enforce Eqs. (2.34) and (2.37) using an NTS query. I describe how to synchronize safely in Section 2.5.2. The receiver must enforce Eq. (2.44) on any adjustment suggested by an NTS query. The conditions proven in this section are restated in Algorithm 2.2 for the convenience of those not concerned with safety proofs.

### 2.5.1 Certifying GIC Safety for Receipt Safety

I start with bounding the clock drift over time $\theta(t)$ to the clock drift at the last synchronization $\theta^l$ with the non-negative, strictly increasing bounding function $B(\cdot)$:

$$|\theta(t) - \theta^l| < B(t - t^l) \ . \tag{2.10}$$

There are two relevant derivation cases, given the absolute value. For the leading case, I have the following, substituting Eq. (2.8):

$$\begin{aligned} \theta(t) - \theta^l &< B(t - t^l) && (2.32) \\ \theta(t) - B(t - t^l) &< \theta^l \\ \theta(t) - B(t - t^l) &< \tau_4^l - t_3^l \\ \theta(t) &< \tau_4^l - t_3^l + B(t - t^l) \ . && (2.33) \end{aligned}$$

To prevent the false alert, I must certify that $\theta(t) < \frac{\Theta}{2}$, but the safe estimate is $\tau_4^l - t_3^l + B(t - t^l)$. Therefore, I desire that the measured bound to be tighter than the safety condition on false alerts, arriving at Eq. (2.34):

$$\tau_4^l - t_3^l + B(t - t^l) < \frac{\Theta}{2} \ . \tag{2.34}$$

For the lagging case, I follow a similar procedure:

$$\begin{aligned} -B(t - t^l) &< \theta(t) - \theta^l && (2.35) \\ \theta^l &< \theta(t) + B(t - t^l) \\ -(t_2^l - \tau_1^l) &< \theta(t) + B(t - t^l) \\ -(t_2^l - \tau_1^l) - B(t - t^l) &< \theta(t) \ . && (2.36) \end{aligned}$$

I must certify that $-\frac{\Theta}{2} < \theta(t)$ to prevent breaking receipt safety. Therefore, once again, I desire that the measured bound to be tighter than the safety condition on security, arriving at Eq. (2.37):

$$-\frac{\Theta}{2} < -(t_2^l - \tau_1^l) - B(t - t^l) \ . \tag{2.37}$$

The receiver can use conditions from Eqs. (2.34) and (2.37) to determine the safety of its GIC at the query time (when $B \approx 0$). Provided an accurate model of $B$, the receiver can determine the time of the next appropriate synchronization. Since $B$ is non-negative and strictly increasing, the receiver can uniquely solve Eqs. (2.34) and (2.37) for the next appropriate synchronization time. That time will be the minimum $t$ between the maximum $t$ satisfying Eq. (2.34) and the maximum $t$ satisfying Eq. (2.37).

One could add Eqs. (2.34) and (2.37) to provide a single safety condition as a function of $l$'s round trip time. However, the separation is helpful for the security proofs since security relies on the lagging bound.

### Proof of Receipt Safety

To show clock synchronization safety in the receipt safety check, I will show that if an adversary delays messages in the protocol, the adversary cannot fool a receiver into believing its unsafe clock is safe. The relevant condition for this proof is Eq. (2.37). The other condition that helps with false alarms is irrelevant to safety in this section. The attack is conceptually diagrammed with Fig. 2.18.

In Fig. 2.18, the zero-latency (i.e., $\varepsilon \to 0$) adversary may elect to introduce a delay $\Delta_{1,2} \geq 0$. Suppose the receiver's clock is broken, meaning $\theta^l < -\frac{\Theta}{2}$. The adversary must select a $\Delta_{1,2} \geq 0$ that passes Eq. (2.37) to fool the receiver into believing its clock is safe.

Starting from the check with Eq. (2.37), I substitute the measurement equation:

$$\begin{aligned} -\frac{\Theta}{2} &< -(t_2^l - \tau_1^l) - B(t - t^l) \qquad (2.37) \\ -\frac{\Theta}{2} &< -(t_2^l - t_1^l - \theta^l) - B(t - t^l) \\ -\frac{\Theta}{2} + t_2^l - t_1^l + B(t - t^l) &< \theta^l \ . \end{aligned}$$

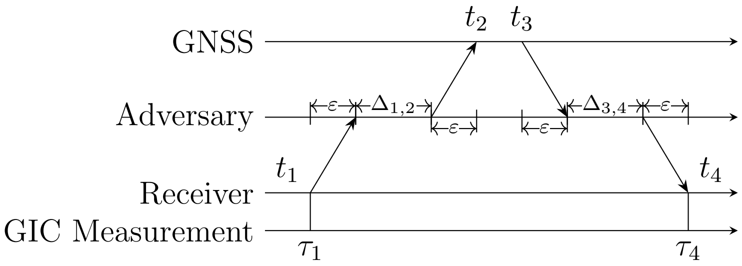


Figure 2.18: Conceptual Diagram Of The Attack Scenario During The GIC Check And Synchronization Scenario.

The adversary may induce delays $\Delta_{1,2}, \Delta_{3,4} \geq 0$ and have zero latency (i.e., $\varepsilon \to 0$). The event numbers come from the NTS protocol.

I substitute the adversarial delay and apply the broken-clock condition:

$$\begin{aligned} -\frac{\Theta}{2} + \Delta_{1,2} + B(t - t^l) &< \theta^l \\ -\frac{\Theta}{2} + \Delta_{1,2} + B(t - t^l) &< -\frac{\Theta}{2} \\ \Delta_{1,2} + B(t - t^l) &< 0 \ . \end{aligned} \tag{2.38}$$

Both $\Delta_{1,2}, B(\cdot) \geq 0$. Being generous to the adversary, I let $B(\cdot) = 0$. Therefore,

$$\Delta_{1,2} < 0 \ . \tag{2.39}$$

This contradicts the initial requirement that the adversary induces $\Delta_{1,2} \geq 0$; therefore, the adversary cannot fool the receiver into believing its broken clock is safe.

From an intuition point of view, when the adversary induces $\Delta_{1,2}$, the adversary causes the receiver to believe its clock lag is worse. As the receiver's lagging clock belief gets worse, the check condition becomes more brittle. This means the adversary's delay will make the alert trigger more sensitive. This argument also applies to $B(\cdot)$.

Suppose an adversary were to induce a $\Delta_{3,4} \geq 0$ on the NTS query return. This manipulation would decrease the upper bound on $\theta$ in Eq. (2.34). But since $\theta$ still adheres to the lower bound of Eq. (2.37), the upper bound is not the active constraint on $\theta$ in the proof of detecting an unsafe clock with the worst $\theta$ under adversarial manipulation.

### 2.5.2 Synchronizing GICs Safely

Suppose that the receiver uses an NTS query to compute a $\delta\theta^l$, which will be the adjustment subtracted from the future output of the clock. This means that if the clock offset were $\theta^l$ before synchronization, it is $\theta^l - \delta\theta^l$ after synchronization. I now derive safety bounds on $\delta\theta^l$ so the clock is safe even with manipulated synchronization.

I start with the safety condition ***after synchronization*** with Eq. (2.40):

$$-\frac{\Theta}{2} < \theta^l - \delta\theta^l < \frac{\Theta}{2} \; . \tag{2.40}$$

Eq. (2.40) enforces that the clock drift after synchronization meets the requirement of Eq. (2.23) for loose-time synchronization. With an NTS query, I can bound $\theta$ via Eq. (2.8). For convenience, I subtract $\delta\theta^l$ from all sides of Eq. (2.8) to arrive at Eq. (2.41):

$$-(t_2 - \tau_1) - \delta\theta^l < \theta^l - \delta\theta^l < \tau_4 - t_3 - \delta\theta^l \; . \tag{2.41}$$

To derive safety bounds on $\delta\theta^l$, I combine Eqs. (2.40) and (2.41).

Eqs. (2.40) and (2.41) each express bounds on $\theta - \delta\theta^l$. However, since Eq. (2.40) is the safety condition the synchronization must satisfy and Eq. (2.41) is a bound on a measurement, the domain of Eq. (2.40) must form a superset of the domain of Eq. (2.41). For the upper side, I derive the following lower bound on $\delta\theta^l$:

$$\begin{aligned} \tau_4^l - t_3^l - \delta\theta^l &< \frac{\Theta}{2} \\ \tau_4^l - t_3^l - \frac{\Theta}{2} &< \delta\theta^l \; . \end{aligned} \tag{2.42}$$

For the lower side, I derive the following upper bound on $\delta\theta^l$:

$$\begin{aligned} -\frac{\Theta}{2} &< -(t_2^l - \tau_1^l) - \delta\theta^l \\ \delta\theta^l &< -(t_2^l - \tau_1^l) + \frac{\Theta}{2} \; . \end{aligned} \tag{2.43}$$

Combining Eqs. (2.42) and (2.43), I form the bound of Eq. (2.44):

$$\tau_4^l - t_3^l - \frac{\Theta}{2} < \delta\theta^l < -(t_2^l - \tau_1^l) + \frac{\Theta}{2} \; . \tag{2.44}$$

Any $\delta\theta^l$ that satisfies Eq. (2.44) will generate a safe clock at the moment of synchronization. Then, Eqs. (2.34) and (2.37) determine how long the clock synchronization is safe over time.

Solving the case where no $\delta\theta^l$ satisfies Eq. (2.44) derives an alert condition as a function of the synchronization's round trip time. To do this, one can flip the direction of the inequality in Eq. (2.44), as in the following:

$$\tau_4^l - t_3^l - \frac{\Theta}{2} > -(t_2^l - \tau_1^l) + \frac{\Theta}{2} \tag{2.45}$$

$$\Theta < \tau_4^l - t_3^l + t_2^l - \tau_1^l \; . \tag{2.46}$$

### Proof of Receipt Safety

To show safety in the synchronization protocol, I will show that if an adversary delays messages in the protocol, the adversary cannot fool a receiver into synchronizing to an unsafe condition. Here, the adversary desires to fool a receiver into selecting a $\delta\theta^l$ so that after synchronization, the clock is outside loose-time synchronization. This corresponds to the condition $\theta^l - \delta\theta^l < -\frac{\Theta}{2}$.

According to Eq. (2.44), the receiver may select any $\delta\theta$ that is in between those bounds (noting that a sensible receiver might elect to select the mid point of the bound). Let's assume that the receiver would choose the most favorable to the adversary. Since the adversary wants $\theta^l - \delta\theta^l$ to be as small as possible, with the negative sign, the relevant bound from Eq. (2.44) is the upper bound. That is, I assume that the receiver selects the largest available $\delta\theta$ to make $\theta^l - \delta\theta^l$ as small as possible to assist the adversary in meeting their win condition. Because the bound involving $\tau_1$ and $t_2$ is the relevant bound, I can reuse the attack diagram from Fig. 2.18. Again, the adversary may introduce $\Delta_{1,2} \geq 0$ and have $\varepsilon \to 0$.

Starting from the unsafe condition, I rearrange for convenience:

$$\theta^l - \delta\theta^l < -\frac{\Theta}{2} \tag{2.47}$$

$$\theta^l + \frac{\Theta}{2} < \delta\theta^l \ . \tag{2.48}$$

Now, I apply the upper bound from Eq. (2.44):

$$\begin{aligned} \theta^l + \frac{\Theta}{2} &< -(t_2^l - \tau_1^l) + \frac{\Theta}{2} && (2.49) \\ \theta^l &< -(t_2^l - \tau_1^l) \\ \theta^l &< -t_2^l + t_1^l + \theta^l \\ 0 &< -\Delta_{1,2} \\ \Delta_{1,2} &< 0 \ . && (2.50) \end{aligned}$$

This contradicts the initial requirement that the adversary introduce $\Delta_{1,2} \geq 0$. Therefore, the adversary cannot fool a receiver into synchronizing into an unsafe condition.

Suppose an adversary were to induce a $\Delta_{3,4} \geq 0$ on the NTS query return. This manipulation would increase the lower bound on $\delta\theta$ in Eq. (2.42). But since $\delta\theta$ still adheres to the upper bound of Eq. (2.42), the lower bound is not the active constraint on $\delta\theta$ in the proof of the smallest obtainable $\theta$ after synchronization under adversarial manipulation.

### 2.5.3 Experimental Validation

In this section, I validate the checks proposed in Sections 2.4 and 2.5 by simulation. To simulate, I generate a representative set of scenarios among $\theta$ and $\Delta$. In my simulations, I set $\Theta = 1$, meaning the forgery condition is $\Delta \geq 1$. I simulate clocks with offsets between -2 and 2. In addition to the adversary-induced delays, I add a

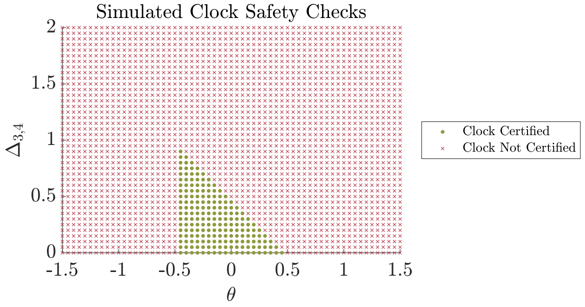


Figure 2.19: GIC Safety Check Validation.

Results from simulations used to validate the clock safety check of Eqs. (2.34) and (2.37). The simulation was conducted using a representative set of $\theta$ and $\Delta$.

0.01 latency within every simulated transmission.

For the proofs related to verifying clock safety and ensuring safe synchronization, the active constraints involve $\tau_1, t_2$ and $\Delta_{1,2}$. This corresponds to proving conditions in the worst case $\theta$ (favorable to the adversary). However, a real spoofer will want to induce a $\Delta_{3,4}$. With $\Delta_{3,4}$, the spoofed measurement will appear to the receiver as if its clock is more ***ahead*** than reality. The spoofer wants a broken lagging clock to believe it is further ahead than reality to fool the receiver of a lagging clock undetected. This means that the adversary inducing a $\Delta_{1,2} \geq 0$ will aid the receiver in detecting its unsafe condition, and inducing a $\Delta_{3,4} \geq 0$ will help the adversary remain undetected when a clock starts lagging to a broken state.

In Fig. 2.19, for each $\theta$ and $\Delta_{3,4}$, I evaluate the check of Eqs. (2.34) and (2.37) by simulation. Under no circumstances is a clock outside the synchronization conditions or in forgery conditions accepted at the time of this check. I note that as the clock offset approaches the lagging unsafe condition, the amount of delay allowed to the adversary before detection increases.

In Fig. 2.20, for each $\theta$ and $\Delta_{3,4}$, I evaluate $\theta$ ***after*** synchronization according to both bounds and midpoint of Eq. (2.44). The receiver's selection of the upper bound is most favorable to the adversary, resulting in the lowest $\theta$ after synchronization. In the figure, when $\Delta > \Theta$, $\tau_4 - t_3 + t_2 - \tau_1 > \Theta$, the receiver knows not to make any changes to the clock and stop authenticating. Hence, for $\Delta > 1$, the simulated clock

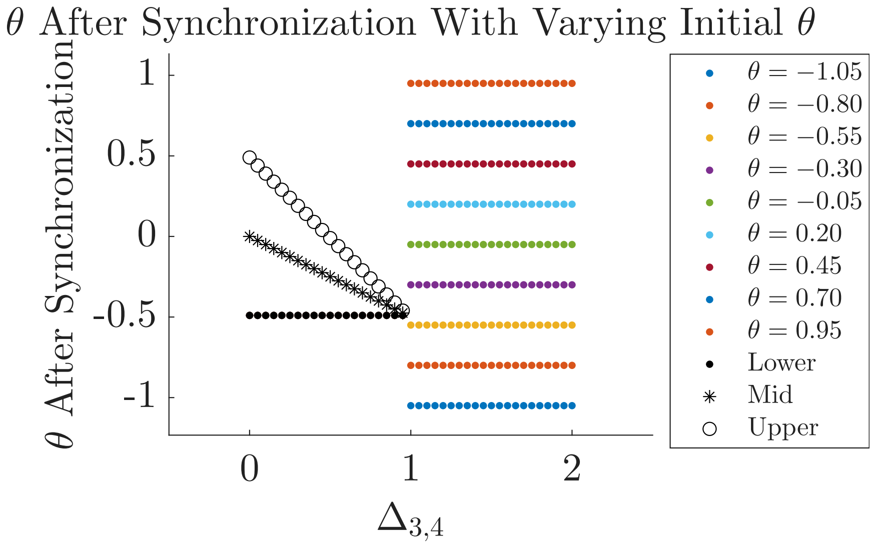


Figure 2.20: GIC Synchronization Validation.

Results from simulations used to validate the synchronization safety of Eq. (2.44). The simulation was conducted using a representative set of $\theta$ and $\Delta$. The markers are black where all the scenarios converge. The dots presume the receiver elects the upper bound of Eq. (2.44) to help the adversary win the game. A sensible receiver will elect to select the midpoint of Eq. (2.44), which are the starred markers. When $\Delta \geq 1$, the adversary may induce forgeries, but the receiver knows not to adjust the clock offset and shutdown; hence, the markers on the right part of the diagram reflect the initially simulated $\theta$.

offsets are not changed, reflecting how they were initially set in the simulations. For $\Delta < 1$, the clocks are set to the same state, which is why they are marked black in that region. The dots correspond to the worst case with the upper bound of Eq. (2.44). The stars correspond to the sensible midpoint of the bounds of Eq. (2.44), which is Eq. (2.3). And the circles correspond to the lower bound of Eq. (2.44). I show each to show that the adversary's manipulation pushes $\theta$ back, which will make the receiver more likely not to detect its lagging condition, but not enough to allow a forgery.

In all cases, I observe what the proofs already show: that these methods will ensure that a receiver correctly ensures receipt safety.

## 2.6 Addressing NTS Vulnerabilities with TESLA-enabled GNSS

Suppose a receiver initiates any of the procedures of Section 2.5 but receives no response from the server. If the receiver does not shut down, the receiver is vulnerable to attack. This brittle shutdown condition could represent a significant inconvenience, and some safety-of-life contexts may prohibit such a shutdown. This vulnerability with TESLA-enabled GNSS results from (1) the broadcast-only context and (2) disclosing or leaking information about $\tau_1$ in the synchronization protocol.

In Section 2.6.1, I describe how vulnerabilities arise and concretely describe how to attack a receiver. In Section 2.6.2, I apply the attack on data provided by an NTP server to show its feasibility. In Section 2.6.3, I describe how to address the vulnerability (but not with the same level of surety as the above proofs).

The vulnerabilities and methods of this section apply to a receiver unwilling to shut down if an attempted time-synchronization fails. After attempting a synchronization, information is leaked about the state of the receiver's GIC. The adversary can use this information to attack unsuspecting, broken receivers. If a receiver is willing to wait for service to check its clock using the methods of Section 2.5, then the receiver will not suffer these vulnerabilities. However, I provide this section because I expect some contexts may need to continue operating when synchronization is denied.

### 2.6.1 Synchronization Vulnerability

Unless NTS is modified according to Section 2.6.3, NTS lends to a vulnerability resulting from (1) the broadcast-only context and (2) disclosing $\tau_1$. First, the loose-time synchronization condition for TESLA security was modified to accommodate broadcast-only GNSS. In ***normal*** TESLA, $\Theta$ is selected based on synchronization pings with Eq. (2.9). If a man-in-the-middle increases $t_2$ or a clock spontaneously increases its lag, $\Theta$ increases to prevent the acceptance of forged messages until there is a denial of service. In the GNSS context, $\Theta$ is constant for the entire constellation. This changes the safety condition to the enforcement of Eq. (2.37). Second, the act of distributing $\tau_1$ reveals information about the state of the receiver's clock. An adversary can use knowledge of a receiver's broken clock to submit forged messages to them and block the unsafe clock's detection.

The mathematical arguments herein impose an assumption on the clock offset over time via Eq. (2.37) and bounding function $B(\cdot)$. However, clocks are stochastic instruments. For instance, radiation could spontaneously flip a register, causing the onboard clock to go into a broken condition that would violate receipt safety. Depending on the clock certification, this would undoubtedly be a rare force majeure, but it still can happen. The principal vulnerability here is that the adversary can identify such broken receivers, and finding just one could cause serious havoc.

The attack is diagrammed with Fig. 2.21, described in Algorithm 2.4, and summarized as follows. An adversary forms a model on the receiver clock's $\theta$ and $\tau_1$ and listens to synchronization traffic. Eventually, a receiver clock will spontaneously drift into violating the loose-time synchronization (when the $B(\cdot)$-clock model fails). When the receiver attempts to synchronize, the adversary will observe the traffic. Using the model and information from the synchronization traffic, the adversary estimates with a high probability that a receiver is non-compliant and blocks the NTS traffic back to that receiver to stop the clock's broken condition detection and correction. The adversary knows the receiver will not shut down with a denied NTS query return. The adversary knows with a high probability that that receiver will accept forged messages.

As an overview of the attack, Fig. 2.21 diagrams the attack with the relevant mathematical quantities. I assume that the adversary and provider are perfectly synchronized to actual time, whereas the receiver may have an offset of $\theta$ that an adversary will hope is $\theta < -\frac{\Theta}{2}$. The receiver will initiate an NTS query and note $\tau_1$ when it sends its initial request. The adversary will observe the NTS request and note its incoming time, which I denote as $t_2^A$. $\varepsilon_{1,2}^{\mathrm{A}}$ is the signal travel time from the receiver to the intercepting adversary. The adversary can allow the NTS request to continue to the provider, but the adversary must block the return journey. If the NTS response is allowed back and then recorded at $\tau_4$, the receiver can correct its clock or know that the round trip time disqualifies using the query and initiates shutdown.

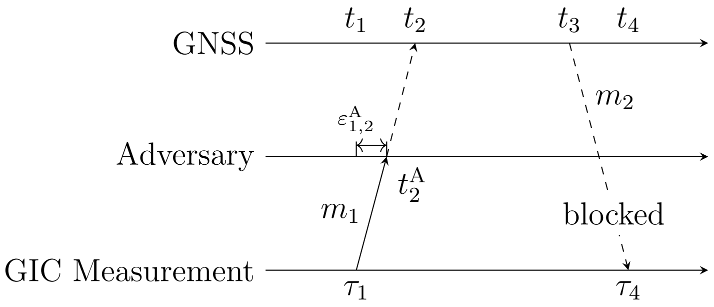


Figure 2.21: Conceptual Diagram Of Attack On Unmodified NTS With GNSS TESLA.

The diagram depicts the structure of the attack on the vulnerability described in Section 2.6.1. $t_1$ through $t_4$ and $\tau_1$ through $\tau_4$ have the same definitions from Fig. 2.5. $\varepsilon^{\mathrm{A}}_{1,2}$ is the network transit time from the receiver to the adversary, and $t^A_2$ is the arrival time to the adversary of that message. As long as the return trip is blocked, the receiver will not be able to have any knowledge about the status of its clock.

**Algorithm 2.4:** Attack On Unencrypted Traffic By An Adversary.

```
Adversary begins observing the time-synchronization traffic of the vehicle
  class associated with a specific location to search for a vulnerable receiver.
Adversary forms a model on the network traffic transit time from the receiver
  to the adversary (e.g., Eq. (2.54)).
Adversary eavesdrops on the NTS request $m_1 = (\eta, \tau_1, s_1^{\mathrm{receiver}})$ where $\tau_1$ is
  the time receiver recorded at the moment of sending $m_1$ and $s_1^{\mathrm{receiver}}$ is an
  authentication signature on $(\eta, \tau_1)$.
Adversary records $t^A_2$, the time of receipt of the eavesdropped message $m_1$.
Using its internet traffic model, $t_1$ and $t^A_2$, Adversary observes traffic until it
  observes a receiver believed to have a $\theta < -\frac{\Theta}{2}$.
Adversary blocks the return NTS response $m_2 = (\eta, \tau_1, t_2, t_3, s_2^{\mathrm{receiver}})$ and
  subsequent return NTS responses back to the receiver believed to accept
  forgeries so that that receiver's clock is never corrected.
Adversary listens to the authentic GNSS signal for a disclosed TESLA hash
  point, generates a forged message and broadcasts it to the vulnerable
  receiver.
```

To derive a model of the attack scenario, I start with the events in the adversary frame with Eq. (2.51). Substituting the measurement equation, I arrive at Eq. (2.52):

$$
\begin{aligned}
t_2^{\mathrm{A}} &= t_1 + \varepsilon_{1,2}^{\mathrm{A}} && (2.51)\\
t_2^{\mathrm{A}} &= \tau_1 - \theta + \varepsilon_{1,2}^{\mathrm{A}} \\
\theta &= -(t_2^{A} - \tau_1) + \varepsilon_{1,2}^{\mathrm{A}} \ . && (2.52)
\end{aligned}
$$

The adversary will use (a) evidence from the NTS traffic and (b) a model of the network traffic transit time to decide whether a specific receiver is vulnerable. In Eq. (2.52), $-(t_2^A - \tau_1)$ is (a), and $\varepsilon_{1,2}$ is (b). For (a), this is a search for outliers. I provide a simple estimation procedure of how long an adversary must search before finding a vulnerable receiver in Section 2.6.2.

For (b), a simple approach will suffice. Using a study of NTP network traffic, such as from [74], I can come up with a correct but somewhat heuristic bound of Eq. (2.53). I justify this bound since NTP round-trip-time is generally around 100 ms and expect the $\varepsilon_{1,2}$ to be around 50 ms, so I consider the model of Eq. (2.53) to be conservative. Nevertheless, an adversary can experientially adjust these considerations to the context:

$$0.99 < \mathrm{Prob}(\varepsilon_{1,2}^{\mathrm{A}} < 100\mathrm{ms}) \ . \tag{2.53}$$

Substituting Eq. (2.52) into Eq. (2.53), I arrive at

$$0.99 < \mathrm{Prob}(\theta < -(t_2^{\mathrm{A}} - \tau_1) + 100\mathrm{ms}) \ . \tag{2.54}$$

Using Eq. (2.54), if an adversary observes traffic where $-(t_2^{\mathrm{A}} - \tau_1) + 100\mathrm{ms} < -\frac{\Theta}{2}$, then that particular receiver is the broken state of $\theta < -\frac{\Theta}{2}$ with high probability.

$\tau_1$ would not be available in the clear if the NTS traffic were encrypted and the server graciously agreed never to leak or disclose $\tau_1$ to malicious parties. Or if the receiver simply omitted $\tau_1$ from the protocol. When $\tau_1$ is not in the clear, the attack model can be modified to incorporate receiver implementation procedures that leak information about $\tau_1$. This would involve replacing $\tau_1$ with $\mathrm{Prob}(\tau_1 \mid t_2^A)$. For instance, if a receiver was known to initiate NTS requests at a specific interval (e.g., the top of the minute), then one could form a model of $\text{Prob}(\tau_1 \mid t_2^A)$ that involves judiciously mapping (e.g., rounding to the nearest minute) $t_2^A$ to likely $\tau_1$. It would be apt for the receiver to introduce randomness to force the adversary to form a ***uniform*** model $\text{Prob}(\tau_1 \mid t_2^A)$ to limit the information gleaned from $t_2^A$.

Solutions that claim to minimize $\tau_1$ information leakage should consider the following scenario. First, the adversary can observe the presence of a successful NTS synchronization and infer that the clock is synchronized to $\theta = 0$ at that time. The adversary has all relevant clock technical specifications, meaning it can compute the next time of synchronization $t$ using the clock's specification of $B(\cdot)$. The adversary could infer that the next $\tau_1$ is the computed next synchronization time $t$ from the GIC specification of $B(\cdot)$. In other words, after observing the last successful synchronization, the adversary can assume that the next synchronization will result from a $\tau_1$ equal to the time derived from its clock drift bound function or some other non-random specification. Moreover, the adversary can block synchronization traffic, forcing the receiver to take worst-case procedures. For instance, an adversary could refuse service to a receiver until the time approaches that last acceptable synchronization time to generate the best possible $\text{Prob}(\tau_1 \mid t_2^A)$. In aggregate, one should assume that the adversary knew the last time when $\theta = 0$ and knows the nominal synchronization time of $\tau_1$.

I should be concerned about the situation where the adversary cares about spoofing ***any*** receiver rather than a particular receiver. In the context of the large vehicles potentially employing TESLA-enabled GNSS (e.g., autonomous cars and planes), finding a single vulnerable vehicle would be enough to wreak havoc. One could assert that, based on the assumption that the clock drift is bounded by $B(\cdot)$, the receiver should know to shut itself off before the clock could ever violate loose-time synchronization. However, I consider the following two situations. First, given the context of flight, force-majeure events will occur, such as radiation spontaneously changing a clock time. Second, the vehicle context will frustrate shut-down requirements (e.g., I expect resistance to conservative requirements that cause flight or ride-share delays resulting from a busy time synchronization server). Therefore, it may be best to implement the mitigations presented in Section 2.6.3.

### 2.6.2 Experimental Observation

Via private correspondence, the authors of [92] graciously gave us access to the $\tau_1$ and $t_2$ values from a 2015 study regarding NTP server usage. The data provided does not contain identifiable information among the NTP users (just the time stamps in the NTP messages). Using that data and a conservative NTP round trip time model, I found large numbers of users that likely had TESLA-non-compliant clocks were they using a future TESLA-enabled GNSS [13]. I emphasize the careful interpretation of the meaning of these results, limiting the conclusion to just that this attack is immediately feasible.

The population of timestamps reflects the varying inconsistencies of users not faithfully implementing the NTP protocol. Among the approximate 12.7 million requests analyzed, about 26% transmitted a null $\tau_1$ (i.e., $\tau_1 = 0$), and about 5% transmitted a coarse integer second $\tau_1$. The mode of the $\tau_1$ centered about the correct time, indicating the majority of users are leaking the lagging state of their clock. While the data was collected in 2015, there were a large number of $\tau_1$ with the calendar year of 1970 (likely from the Unix epoch). Moreover, a substantial section of users form an apparent uniform floor. I suspect this uniform band results from users transmitting a random $\tau_1$, perhaps as a nonce to differentiate repeated requests. Lastly, I note that I observe elevated bursts, such as the top of the hour and top of the minute, which could reveal information about specific requests' $\tau_1$.

I provide a close-up of the distribution of $-(t_2 - \tau_1)$ values around the correct time in Fig. 2.22. I find a substantial number of users who, ostensibly, are providing a faithful $\tau_1$ value, and I observe a substantial incidence of users whose clocks are likely lagging behind. In Fig. 2.22, I annotate which users, if they were listening to a future authenticated SBAS concept with $\Theta = 6$ [13], would accept forgeries.

From these results, I make the following ***limited*** claim. Since I used data provided by the back-end of an NTP server to observe clocks of users who would, with high probability, accept forgeries in a TESLA-enabled GNSS were they using that TESLA-enabled GNSS, the NTP server would certainly be able to identify these vulnerable users. Moreover, given the threat model and how the traffic is unencrypted, an adversary would also be able to identify vulnerable receivers immediately. The

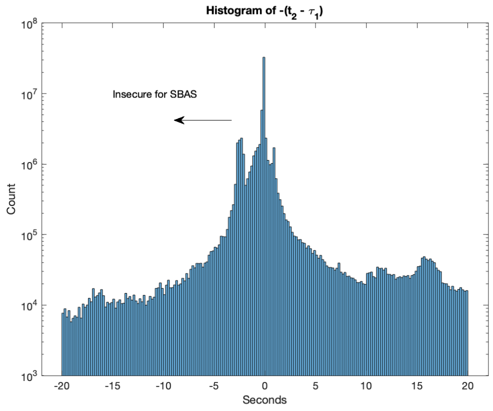


Figure 2.22: Histogram Results From NTP Study.

The distribution of $-(t_2 - \tau_1)$ zoomed into the mode of the distribution around near-synchronized clocks. Neglecting a conservative 100ms NTP request transmission time, a negative value indicates that a particular clock is lagging. If the user were to use a TESLA-enabled GNSS, a value less than $-\frac{\Theta}{2}$ indicates that the user would accept forgeries. I annotate where future TESLA-enabled SBAS users of [13] would accept forgeries.

adversary's game is to wait for a vulnerable user. I show the concrete plausibility of exploiting the vulnerability I point out. Therefore, this vulnerability should concern those wishing to provide authentication security via a TESLA-enabled GNSS.

I must limit the interpretation beyond a demonstration. Interpretation of the actual data should be limited given the wide variety of actual uses of $\tau_1$. Moreover, I do not warrant that this traffic is representative of future synchronization traffic for a future TESLA-enabled GNSS. I would have no indication whether the time $\tau_1$ is measured from the clock used to enforce the TESLA-loose-time synchronization requirement or a separate clock. Furthermore, I do not expect the drift rates of receiver clocks to be consistent with the drift rate of clocks from this NTP traffic.

#### Poisson Model

An adversary must wait to find a vulnerable clock. Depending on the receiver clock drift rate, the incidence might be similar to that of rare outliers. However, by characterizing the data, an adversary can reasonably model how long they will expect to wait, such as with a Poisson model.

Suppose this data came from users using a TESLA-enabled SBAS with $\Theta = 6$, as in [13]. Considering the wide variety of $\tau_1$ observed, I assume incoming NTP requests received within 20 seconds of real-time contain $\tau_1$ that reflects their real clock. Therefore, among those requests, requests with a $-(t_2 - \tau_1) < -3.1$ should accept forgeries with high probability. Taking the $t_2$ from those offending requests, I form a maximum-likelihood-estimated Poisson model on the incidence of unsafe receivers using the NTP service. I calculate a Poisson model with $\lambda = 0.57$ seconds. Therefore, on expectation, for this data, the adversary should expect to encounter a vulnerable clock after 0.57 seconds of observation. There were about 7000 requests per second in this data set, which handily suggests that vulnerable clocks are certainly outliers. Moreover, I also note that I expect vulnerable clocks among those who did not send a faithful $\tau_1$.

### 2.6.3 Addressing NTS Vulnerabilities

To mitigate the threat posed by Section 2.6.1, a receiver must limit information leakage of $\tau_1$. This can be done with two simple modifications to NTS. First, $\tau_1$ should be omitted from NTS messages or replaced with a nonce. Second, NTS queries should occur uniformly randomly over a large enough interval.

Regarding the first modification, I already observed users doing this in Section 2.6.2. Removing $\tau_1$ from the protocol poses no burden to the synchronization because the non-receiver party does not need $\tau_1$ in any computation. If one were to encrypt the NTS traffic, an eavesdropper would not be able to engage in this attack; however, the synchronization server could leak $\tau_1$ or use it as an adversary itself. In Section 2.6.2, the server did this so I could demonstrate this attack. Therefore, it would be best to not trust the server and simply omit $\tau_1$ or replace $\tau_1$ with an

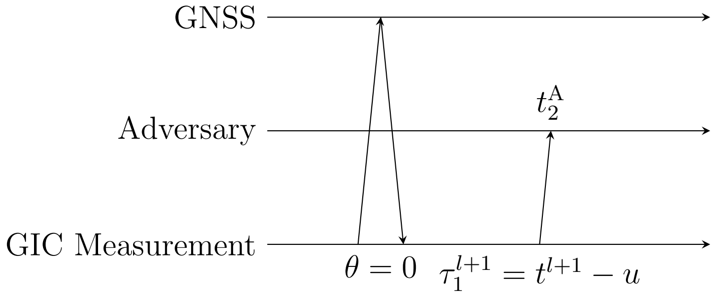


Figure 2.23: Conceptual Diagram Of Attack Where The Adversary Estimates $\tau_1$.

The adversary uses the time of the last synchronization to predict $\tau_1$ to form a model of $\mathrm{Prob}(\theta \mid t, t_2)$. To mitigate this attack and limit the information leakage from synchronization itself, I suggest the receiver introduce uniform randomness into the NTS query initiation. This ensures that an adversarial model of $\mathrm{Prob}(\theta \mid t, t_2)$ must be uniform. The transmission lines approach vertical to reflect the case when the adversary is immediately adjacent to the receiver to achieve a conservative case and simplify math.

unrelated value.

Regarding the second modification, NTS queries that occur on a predictable basis leak information about $\tau_1$. A good counter-action to this security concern is to ensure that the best model an adversary can form on $\mathrm{Prob}(\tau_1 \mid t_2^A)$ is a uniform distribution with support exceeding anything useful to the adversary. A uniform $\mathrm{Prob}(\tau_1 \mid t_2^A)$ translates to a uniform adversarial-deduced distribution of $\theta$ given the evidence from adversarial eavesdropping.

As in the conceptual diagram of Fig. 2.23, suppose that the last synchronization occurred at $t^l$ and, using Eqs. (2.34) and (2.37), the receiver computes the next $t$ for synchronization. Suppose the receiver initiates the next synchronization at $\tau_1^{l+1} = t - u$ with $u \leftarrow U(0, 2\lambda\Theta)$. $\lambda \geq 1$ is a slack parameter that indicates how much support I will allow with the adversarial model. $\lambda = 1$ corresponds to the case where the spread is equal to the allowed drift assuming $B(\cdot)$ in between synchronizations.

I show that, given the adversary-observed evidence, the best model on $\theta$ is uniform. Suppose that the adversary is immediately adjacent to the receiver, meaning the transit time from receiver to adversary is instantaneous, forming a conservative model while simplifying math. With Eq. (2.55), I derive Eq. (2.56) by substituting the receiver's selected resynchronization time from the preceding paragraph.

$$\begin{aligned} t_1^{l+1} &= t_2^{l+1} \qquad (2.55) \\ \tau_1^{l+1} - \theta^{l+1} &= t_2^{l+1} \\ t - u - \theta^{l+1} &= t_2^{l+1} \\ \theta^{l+1} &= t - t_2^{l+1} - u \; . \qquad (2.56) \end{aligned}$$

From Eq. (2.56), $t - t_2$ is known to the adversary. Since $u$ is uniform and unknown to the adversary, the distribution $\text{Prob}(\theta^{l+1} \mid t - t_2^{l+1})$ is uniform over $(t - t_2^{l+1}, t - t_2^{l+1} - 2\lambda\Theta)$.

My suggestion suffers from two issues. First, it's a necessity after several possibly assumed-true conditions are false. These include (a) that the clock spontaneously drifted outside the already assumed bound $B(\cdot)$ and (b) the receiver continued to operate beyond its resynchronization specification after being denied service by NTS. However, I note that this entire discussion is about identifying outlying and adverse events for contexts that may not have the option to shut down (e.g., moving or flying vehicles). Second, my suggestion does not provably assure no information leakage: consider the case when the next NTS ping comes so egregiously late, ensuring that the entire spread of uniform $\text{Prob}(\theta^{l+1} \mid t - t_2^{l+1})$ is in a broken state. As the clock approaches the boundary $\theta < -\frac{\Theta}{2}$ and the support of uniform $\text{Prob}(\theta^{l+1} \mid t - t_2^{l+1})$ includes some broken-clock domain, an adversary can compute the probability that a receiver is broken given the evidence.

A receiver could choose a $\lambda = \frac{t}{2\Theta}$ so that a receiver is always uniformly querying their clock, producing a $\text{Prob}(\theta^{l+1} \mid t - t_2^{l+1})$ with the maximal support confusing the adversary. However, an adversary could block all NTS attempts until the clock nears its specification time $t$, undoing a large-uniform-support strategy. I suspect there is no provable way to initiate an NTS query that admits no information leakage on $\tau_1$ with a single GIC. To provide provable safety against this attack, a receiver must shut down if an NTS query ever fails.

# Chapter 3

# GNSS TESLA Design

> You can't just ask customers what they want and then try to give that to them. By the time you get it built, they'll want something new.
>
> Steve Jobs

The principal advantages of using Timed Efficient Stream Loss-tolerant Authentication (TESLA) over exclusively using asymmetric digital signatures with Global Navigation Satellite System (GNSS) are more efficient data bandwidth and loss tolerance for authentication [63]. This chapter is about how to leverage TESLA cryptographic constructions for GNSS authentication[1]. Before moving on to more complex TESLA constructions for GNSS, this initial section discusses an overview of the data bandwidth efficiency gains possible with TESLA to motivate the rest of this chapter. In the introduction from Section 1.3.4, each message received its own commitment-MAC tag (CMT) and hash point. For this section, I will call that naive implementation out-of-the-box TESLA, and this chapter explores how to improve upon out-of-the-box TESLA.

Each TESLA hash path must also include an asymmetric digital signature (DS),

[1] This chapter is based on my two publications regarding efficient TESLA constructions for GNSS [7, 8].

as discussed in Section 1.3.4. However, this use can be better thought of as infrequent maintenance. For instance, with a proposed TESLA scheme for the Satellite-based Augmentation System (SBAS) from [13], a message containing a hash point is transmitted every six seconds, and the DS information is transmitted every five minutes. Chapter 4 covers the infrequent maintenance transmission in depth, but for this section, one must remember to account for that maintenance when doing a fair comparison of TESLA and a DS-only scheme. Table 3.1 provides a comparison of data bandwidth requirements of a DS-only scheme and several TESLA schemes. In this section, I will assume that the DS protocol is Elliptic Curve Digital Signature Algorithm (ECDSA); however, the general observations are still the same when using other protocols (e.g., EC-Schnorr, quantum signature algorithms).

From Table 3.1, first, let us compare the first and second rows. Table 3.1 assumes a message each second and that the maintenance data must be transmitted every five minutes from [13]. In the first row, I provide the data required for an ECDSA-only scheme. In the second row, I provide the data required for the out-of-the-box TESLA-ECDSA scheme. While the out-of-the-box TESLA requires more maintenance data than the ECDSA-only scheme because less data is needed to authenticate each message, there is already a near-double improvement of the data bandwidth for authentication.

In the GNSS context, higher missed detection probabilities on the order of $10^{-9}$ are acceptable. To alleviate the data bandwidth from large CMTs, they are usually truncated to around 32-bits ($2^{-32} < 10^{-9}$). This motivates row three of Table 3.1, a GNSS modification of the out-of-the-box TESLA.

Out-of-the-box TESLA comes with some useful loss-tolerance properties over ECDSA. With out-of-the-box TESLA, lost hash point bits can be recovered from the next released hash point. With an ECDSA-only scheme, if a single message or ECDSA bit is lost, the message cannot be authenticated. This is a material concern for satellite communication systems. Due to data bandwidth concerns, [70] suggested that SBAS sign five messages at a time. To further improve the performance in lossy conditions, [70] suggested that each TESLA message include five separate smaller CMTs. The separation enables some authentication even when some messages are

| Protocol | Authentication Bits | Total Per Message |
|---|---|---|
| ECDSA 128-bit Security | *Maintenance:* 256 bits<br>ECDSA Public Key: 256 bits<br>*Per Message: 512 bits*<br>ECDSA Signature: 512 bits | 512.9 bits |
| TESLA 128-bit Security<br>Out of the box | *Maintenance:* 1024 bits<br>ECDSA Public Key: 256 bits<br>hash path end (HPE): 128 bits<br>salt: 128 bits<br>ECDSA Signature: 512 bits<br>*Per Message:* 256 bits<br>hash point: 128 bits<br>HMAC: 128 bits | 259.41 bits |
| TESLA 128-bit Security<br>$10^{-9}$ Missed Detection<br>4 Message Groups | *Maintenance:* Same as above<br>*Per Four Messages:*<br>hash point: 128 bits<br>4 HMACs: 128 bits | 67.41 bits |
| TESLA 128-bit Security<br>$10^{-9}$ Missed Detection<br>5 Message Groups<br>With Erasure<br>Recovery [108] | *Maintenance:* Same as above<br>*Per Five Messages:*<br>hash point: 128 bits<br>aMAC[108] Data: 84 bits | 48.21 bits |

Table 3.1: Comparison Of Data Bandwidth With An ECDSA-only And Several TESLA With ECDSA Schemes.

The table lists and compares all of the authentication data and maintenance data required to authenticate data in a message stream. The total per message presumes the case for SBAS from [13] where a message is transmitted each second and the maintenance data must be transmitted every five minutes. Transmitting the maintenance data every five minutes enables a receiver to complete a first authentication from a cold start within five minutes (but otherwise the time to authentication (TTA) is shorter for warm start and approaches $\Theta$). Moreover, I assume that the receivers have an authenticated commitment of the ECDSA public key pre-stored on the device, as is the case with Galileo's Open Service Navigation Message Authentication (OSNMA).

lost. This idea was further improved in [108] with an erasure recovery strategy to provide a $10^{-9}$ missed detection probability and allow two out of each five messages to be lost, motivating row four of Table 3.1. From row one to row four in Table 3.1, there is a tenfold improvement of data bandwidth for authentication, and I will go further in this chapter.

To go from row two to row three or four in Table 3.1, the GNSS designer must ***judiciously*** use key-derivation function (KDF). Section 3.1 explores what can be done and how to ensure correctness. Section 3.2 explores leveraging the constellation's satellites as parallel and redundant channels to achieve better authentication data bandwidth efficiency. Lastly, Section 3.3 applies the principles of this chapter to create some example authentication design sketches to various GNSS application signals.

## 3.1 KDF Geometry

In Section 1.3.4 where I introduce TESLA, the diagrams starting with Fig. 1.6 start to follow a recurring format. Each diagram reads from left to right with increasing time of release, and there are several one-way cryptographic operations represented with arrows. In Fig. 1.8, there are two one-way operations: the first relates individual hash points, and the second relates hash points with CMTs. The arrows point in the opposite direction of time. Because each one-way arrow points in the opposite of time, the scheme is secure.

[41, 70] suggested having one hash point sign multiple messages for loss tolerance purposes. In this work, I will use one hash point to sign and generate a large amount of key material, enabling more features with limited additional burden on the data bandwidth. A nuanced difference to prior art: rather than having satellites cross-authenticate each other, satellites ***integrally*** share the entire TESLA cryptographic structure and rely on that structure to decrease TTA (and time to first authenticated fix (TFAF) in Chapter 4). But first, to ensure authentication security, one must understand a more complex set of geometric rules to ensure security.

Fig. 3.1 provides a conceptual diagram where a single hash point signs multiple messages in a stream. To mitigate related key attacks and design implementation

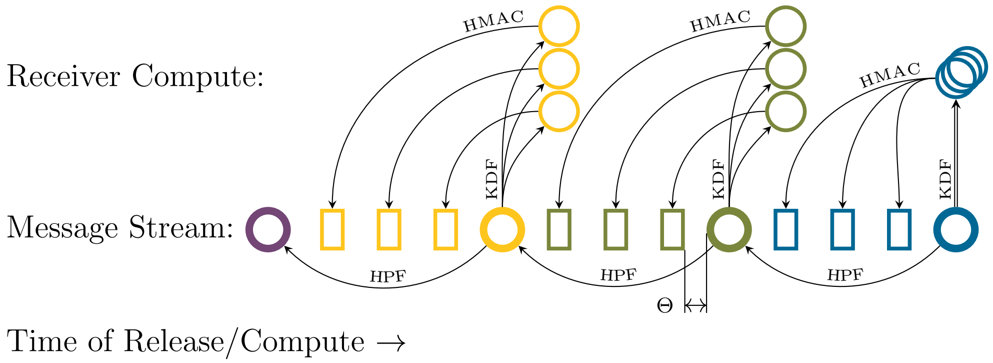


Figure 3.1: Conceptual Diagram Of Using KDF To Derive Multiple Keys To Sign Multiple Messages In A Message Stream.

The diagram includes two row groups labeled Message Stream and Receiver Compute. The bottom row group is similar to earlier diagrams, such as Fig. 1.7. The top row group represents additional derived keys that are ***not*** distributed in the message stream. Rather, the receiver derives them via KDF with Eqs. (3.1) and (3.2). The derived keys are represented to the right of the corresponding hash point because the receiver may only compute them after the hash point release. The right-most iteration is the same as the others, but it is shown abbreviated. This abbreviation will be common in future diagrams. Moreover, the right-most's derived key horizontal positions reflect the simplification resulting from assuming a zero-latency adversary. Because these derived keys do not burden the data bandwidth of the message stream, the GNSS designer can generate numerous derived keys to sign many different pieces of information. This geometry is secure because the security condition is met. Consider the earliest-released object from each hash point to message path. There is a one-way operation ***against*** time from ***that*** object to the message with a horizontal distance ***at least*** Θ.

mistakes, each context (i.e., message or other pseudorandom material, such as the watermarks from Chapter 5) should receive its own independent key. To accomplish this, I introduce an ***intermediate*** KDF operation in between the hash point and any final authentication operations. With math, this is represented by the following:

$$k_{\mathrm{CMF},i,\mathrm{context}} = \mathrm{KDF}(p_i, \mathrm{context}) \tag{3.1}$$

$$\mathrm{CMT}_{i,\mathrm{context}} = \mathrm{CMF}(k_{\mathrm{CMF},i,\mathrm{context}}, m_{i,\mathrm{context}}) \; . \tag{3.2}$$

The intermediate KDF ensures that all CMTs and other pseudorandom material are based on cryptographically ***independent*** cryptographic material.

As long as the context from Eq. (3.1) is unique, each CMT key will be collision-resistant. In Fig. 3.1, the collision-resistance property of each KDF one-way arrow enables each circle to be represented separately. The construction resulting from Eq. (3.1) is simple for the GNSS designer and mitigates a plethora of niche attacks. For instance, when an adversary spoofs a repeated message or engages in related-key attacks [20, 25, 81]. If the GNSS designer needs additional cryptographic pseudorandom material (without imposing a burden on the message stream data bandwidth), they need only introduce another KDF from Eq. (3.1) with a new context.

The context can be any data that is unique for its use. For instance, in the case of SBAS from [13], a single hash path can sign all satellites and all frequencies simultaneously. A good context would be the concatenation of each context (e.g., context $=$ PRN$||$Frequency$||t_i$). Since there is exactly one message per satellite, frequency, and time, each CMT key will be independent, and each message CMT will be different (subject to the truncation collision resistance strength) even if messages are the same.

In the Receiver Compute row of Fig. 3.1, each of the three derived CMT keys are depicted to the right of (or above, in the right-most iteration) the original hash point. This represents how these derived keys are not released in the message stream; instead, the receiver computes them after the release of the hash point via Eq. (3.1). In the last set of messages in **blue**, the derived keys are represented directly above the released hash point. This corresponds to the zero-latency adversary in the context of the security condition imposed on the geometries of this work. While the derived keys are not directly released in the message stream, they are effectively released simultaneously as the hash point.

Suppose that one can generate a diagram where all of the derived information is represented horizontally with release time, as in Fig. 3.1. Release time means the ***earliest*** time at which ***anyone*** beyond the GNSS provider knows the information. The GNSS designer can employ any geometry provided the geometry meets the following security condition on each path from each hash point to each piece of authenticated information. Tracing from each hash point to every piece of authenticated information, consider the object (e.g., a derived key, cryptographic seed) that has the earliest released time. The operation from that earliest object to the authenticated information must be one-way in the opposite direction of time and the horizontal distance from the authenticated information's last bit to that earliest object's first bit must be at least Θ.

In Fig. 3.1, in the left and middle iterations, the derived keys are depicted after the corresponding hash point, representing the delay from the release of the hash point to the actual receiver computation of the receiver. In the hash-point-to-authenticated-information paths, the earliest piece of information is the hash point itself, making satisfying the security condition of the previous paragraph simple. In the right-most iteration, the derived keys are depicted above the hash point, corresponding to the zero-latency adversary (who counts as anyone in the context of the release time definition). Therefore, in the hash-point-to-authenticated-information, the earliest piece of information is the hash point itself or any of the derived keys, and they all meet the security condition in Fig. 3.1.

## 3.1.1 Multi-cadence Distribution

This section discusses constructing geometries that can accommodate multiple user groups with varying access to network connections. These network connections can enable those users to have faster TTA. In this section, I examine two geometry groups: non-diverging and diverging.

### Non-Diverging Geometries

For non-diverging, consider the geometry of Fig. 3.2. In Fig. 3.2, there is a single hash path. Each hash point and derived key is represented by a circle, and each message and CMT combination is represented by a rectangle. In the geometry, a hash point is released via GNSS every 12 messages. The colors correspond to which hash point and derived keys correspond to which messages, and the minimum delay Θ and maximum TTA are marked. Naively, one could have the last hash point sign each of the 11 messages; however, Fig. 3.2 presents a geometry with some advantages.

In Fig. 3.2, Network A and GNSS take turns releasing the hash points: Network A

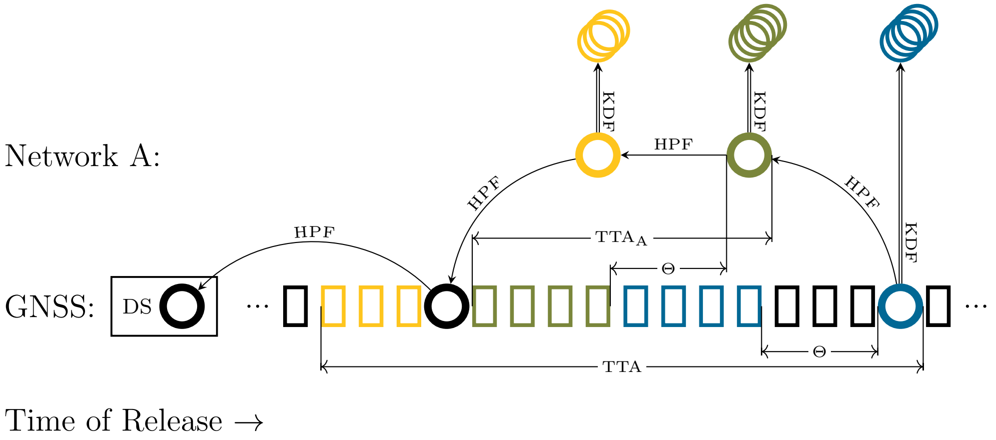


Figure 3.2: Conceptual Diagram Of Non-diverging Multi-cadence Distribution.

Compare to Fig. 3.3. In the diagram, a hash point is released via GNSS every 12 messages. One could have the last (**blue**) hash point authenticate the entire set of 11 messages, corresponding to the TTA and Θ marked below the message stream. However, the GNSS designer could instead design the system with the understanding that some users will have access to Network A. In the diagram, Network A and GNSS coordinate to release hash points at triple the rate, where GNSS only releases one every three. The disconnected receiver can authenticate at the same cadence as before by deriving the Network-A hash points it never received, but the connected receiver enjoys a shorter TTA.

releases two, and then GNSS releases a third, and so on. Receivers without access to Network A can still authenticate every message because the two hash points released by Network A are derivable from the later one distributed via GNSS. Receivers with access to Network A can authenticate with a shorter TTA.

Several miscellaneous notes for the geometry of Fig. 3.2 include the following, and the general principles apply to all geometries using this technique. First, the time synchronization requirement Θ is the same for both types of users, even though receivers with Network A will operate at a faster authentication cadence. Whenever the network and non-network receivers use the same distributed CMTs (e.g., as in Fig. 3.2 where the Network A CMTs are still distributed in the GNSS stream), the time synchronization requirement Θ is the same for both user groups. Second, the receiver must perform two additional hashing operations in its authentication loop over the naive geometry. Third, when generating a new hash path, GNSS must

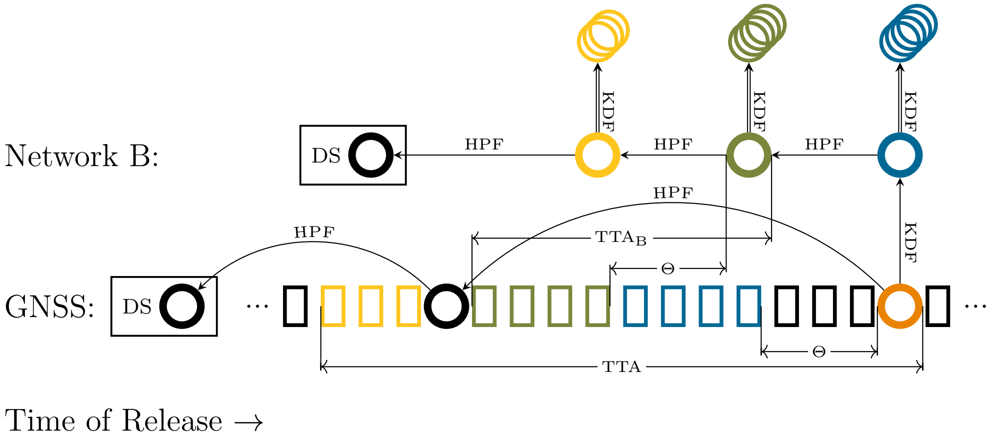


Figure 3.3: Conceptual Diagram Of Diverging Multi-cadence Distribution.

Compare to Fig. 3.3. In the diagram, a hash point is released via GNSS every 12 messages. One could have the last (**orange**) hash point authenticate the entire set of 11 messages, corresponding to the TTA and $\Theta$ marked below. However, the GNSS designer could instead design the system with the understanding that some users will have access to Network B. In the diagram, GNSS computes a diverging path from the original hash path and Network B releases both a DS on the end of the diverging path and the diverging path itself in reverse order. The disconnected receiver can authenticate at the same cadence as before by deriving the Network-B hash points it never received, but the connected receiver enjoys a shorter TTA.

perform three times the hashing operations. Fourth, in the construction of Fig. 3.2, the first **green** colored hash point only has three derived keys instead of four. There is no need to redundantly sign a hash point with CMT because the preimage argument already authenticates it.

### Diverging Geometries

For diverging, consider the geometry of Fig. 3.3. From a receiver perspective, nearly all of the performance indicators are the same in Fig. 3.3 as in the non-diverging case of Fig. 3.2. For instance, the TTA for Network B users is the same as those using Network A. In Fig. 3.3, the hash path repeatedly diverges, and the derived keys derive from the diverged paths. However, there are several important network bandwidth and computational differences.

First, recall that the authentication security argument requires that a receiver verify that the incoming hash point is a preimage of a hash point signed with a DS. Network B ***must*** distribute a DS for ***every*** diverging path end, and the Network B receiver must authenticate the diverging path with these additional DSs. Otherwise, there is no TTA advantage. This follows from TESLA authenticity argument that each preimage hash point must hash down to a hash point with a DS. Without these additional DSs over Network B, a Network B user must ***wait*** until it receives a GNSS-distributed hash point (that already hashes down to a hash point with a DS) nullifying the purported TTA advantage. A non-Network B receiver does not need to have nor verify the DS of diverging paths because the GNSS-received hash point acts as a loss-tolerant distribution and hashes to the DS already.

Second, utilizing a diverging geometry decreases the computation needed by the GNSS provider when generating a new hash path compared to the non-diverging geometry. When creating a new hash path, the GNSS provider need only compute the entire original path before distributing the DS on the HPE. Each diverging path can be computed before use (but before distribution of the Network B DSs). This computation consideration does not have material effects in the simple cases of Figs. 3.2 and 3.3. However, diverging geometries become necessary when the ranging code derives from the hash path due to computational concerns (see Section 3.1.2).

Additionally, I note that hash path function (HPF) should be used along the diverging path because it is salted, providing a ***hiding*** commitment function. One could use KDF provided it has the salting needed to provide the hiding property. In this section, KDF does not include the needed salting to serve as the function along the diverging paths.

### Combinations and Time Synchronization

GNSS is free to combine diverging and non-diverging geometries to suit a wide breadth of segmented (potentially revenue-producing) receiver groups. Fig. 3.4 provides a construction that combines the non-diverging and diverging geometries of Figs. 3.2 and 3.3 and introduces a third diverging geometry. At present in Fig. 3.4, the Network A and B geometries provide redundant TTA performance features. However, if the

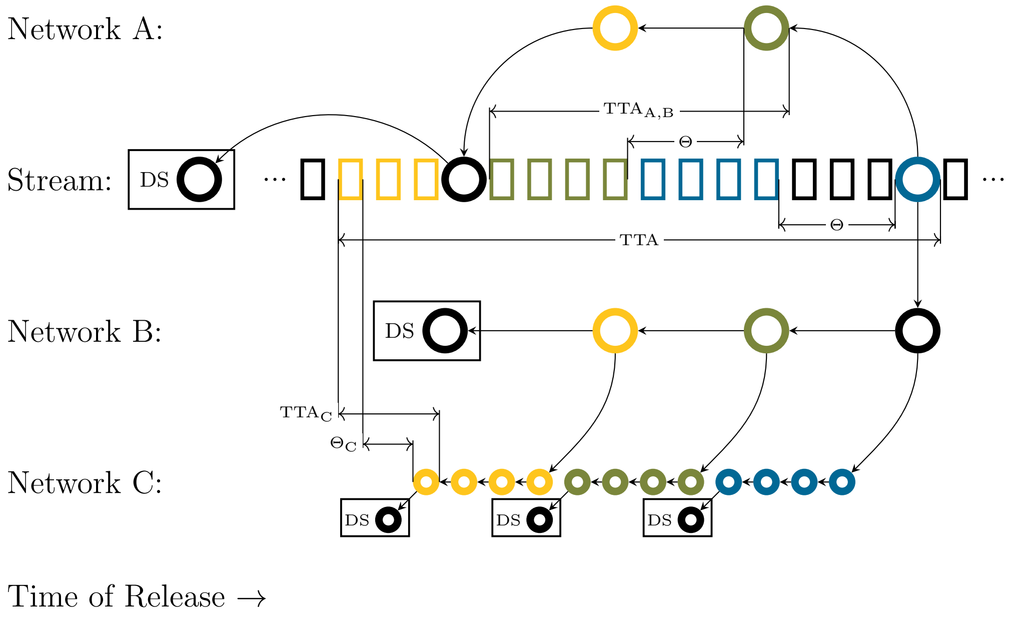


Figure 3.4: Conceptual Diagram Of Multiple Non-Diverging And Diverging Geometries And Time Synchronization.

The GNSS designer is free to include non-diverging and diverging geometries (see Figs. 3.2 and 3.3) provided that the time synchronization is correctly accounted for. The diagram does not depict whether the network CMTs are concurrent for the entire constellation or if they are independent and distributed via the networks. If any of the network's CMTs are concurrent (e.g., Network C's CMTs are the only CMTs distributed over the original stream), then the smallest Θ is the applicable one. If the network's CMTs are only distributed via the network, or there are multiple sets of CMTs distributed over the original stream, then the time synchronization requirements can remain separate.

designer shifts the Network B distributions left, then $\mathrm{TTA}_{\mathrm{B}}$ and $\Theta_{\mathrm{B}}$ shrink. Then, the GNSS could provide three concurrent services for the original stream and Networks A, B, and C: for instance, a free original GNSS stream and a silver, gold, and platinum subscription service for Networks A, B, and C, respectively. The GNSS provider could charge a premium for faster authentication times or greater network use (e.g., Network B needs more transmitted data than Network A).

Fig. 3.4 does not depict where the CMTs are distributed. They could be distributed within the original stream or over the respective networks. This design decision has an important consequence for time synchronization.

In Fig. 3.4, $\Theta_{\mathrm{C}}$ is smaller than the original $\Theta$. If the CMTs derived by the Network C diverged path are distributed in the original stream, then the constellation's $\Theta$ is actually $\Theta_{\mathrm{C}}$. Receivers with a synchronization compliant with $\Theta$ are susceptible to the attack from Section 2.4.4. If the Network C CMTs are distributed over Network C, then the synchronization requirements for the original and Network C receivers can be different. The same principle applies to a shifted Network B geometry.

### 3.1.2 Ranging Code Generation

There are two methods to implement ranging authentication. In both methods, the ranging code includes a hiding bit commitment within the TESLA protocol to provide authentication. Fig. 3.5 provides a conceptual diagram of the two methods: pseudorandom function (PRF) Ranging and Watermark Ranging.

**PRF Ranging**

GNSS broadcasts a pseudorandom sequence to provide a Positioning, Navigation, and Timing (PNT) service. The receiver uses a replica of the pseudorandom sequence to complete signal processing on the signal to measure its range with each satellite. The pseudorandomness ensures that the sequence has high autocorrelation and low cross-correlation, enabling the receiver to lift multiple ranging measurements out of the noise. While a considerable amount of effort has been completed to finding mathematical sequences that fit the low-cross-correlation property and other properties (e.g., ease of computation) [66, 89, 94], simply using a sequence generated using a cryptographic PRF can suffice.

When pseudorandom cryptographic information derives from a TESLA hash path of reverse-released hiding commitments, the pseudorandom cryptographic information appears unpredictably random to the adversary. When using the output of a PRF with a random key as the ranging code, the output is uniformly random (until the delayed release of the hash point and until the security level of the underlying cryptographic primitive). Therefore, the tail bounds on cross correlation between

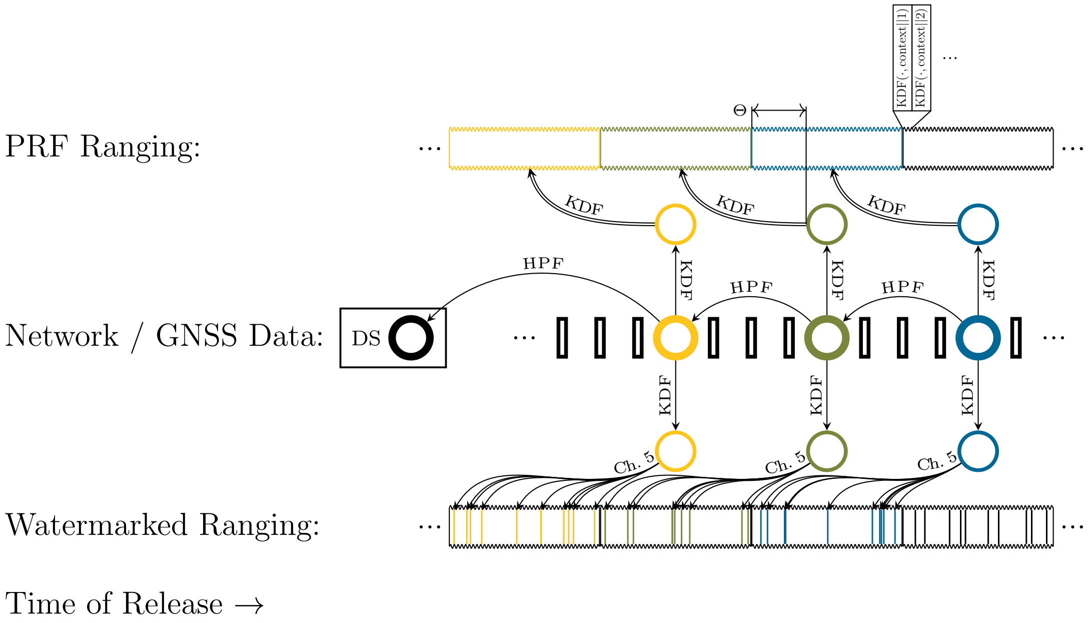


Figure 3.5: Conceptual Diagram Of PRF And Watermarked Ranging Code Generation.

In the top signal, the entire ranging code is generated using a set of parallel KDF operations from the hash point. Because the receiver cannot track the KDF signal until the release of the hash point, the hash point must be distributed via network or another GNSS signal. In the bottom signal, GNSS distributes a known ranging code but incorporates a watermark derived from the hash path as the commitment under TESLA. Presuming watermark marginally degrades the ranging code, the receiver can still track the signal. The receiver must complete additional signal processing after distribution of the hash point to assert security (see Chapter 5).

two different PRF outputs can easily be derived and are favorably exponentially small. While other ranging codes (e.g., Gold Codes) provide a nice, deterministic cross-correlation bound, the PRF bound is functionally equivalent. Moreover, the cryptographic guarantee provides that an adversary would need inordinate computational resources to defeat the favorable exponential bounds. PRF ranging codes are used everyday with the Global Positioning System (GPS) P(Y) and M-Code signals and Galileo's Public Regulated Service (PRS) signal.

Within the TESLA framework, KDF can be used to generate practically infinite ranging codes, just like KDF on the hash point can be used to authenticate additional items in the message stream. This is achieved by simply incrementing the contextual input to KDF, as in the top signal of Fig. 3.5. In Fig. 3.5, a network (or a GNSS data stream of ***another*** pilot signal) provides the hash point distributions from which a PRF ranging code derives.

Because of the nature of PNT, a receiver cannot track a pure PRF signal. To utilize a pure PRF signal, the receiver must receive the hash points via another channel. This results from the receiver not having the PRF replica for signal processing. Interestingly, the act of ranging with a pure PRF ranging code provides authentication. This results from the adversary also not knowing the PRF ranging code until hash point distribution. For instance, a dual-purpose (i.e., data and ranging) signal (e.g., GPS C/A) where the hash points are in the data channel and the ranging channel is a pure hash-point-generated PRF ranging code would not function. Without tracking the signal, the receiver cannot receive the hash points; without the hash points, the receiver cannot compute the PRF ranging code replica to track the signal. The GNSS designer must enable a network to deliver the hash points, include a pilot signal to deliver the hash points, or incorporate a watermark into a known non-cryptographic ranging code.

### Watermarked Ranging

Since the receiver cannot track a PRF ranging code without the assistance of another channel, the GNSS designer may consider watermarked ranging. Fig. 3.5 depicts watermarked ranging with the bottom signal.

As depicted in Fig. 3.5, KDF can generate additional cryptographic seeds from which a watermark is applied to a known non-cryptographic ranging code. The watermark should not degrade the ranging code too much so that the receiver can still track the signal without knowledge of the hidden watermark commitment. Chapter 5 discusses an ideal function by which a non-cryptographic ranging code can be watermarked with an input seed. Unlike with PRF ranging, the act of ranging with a watermarked ranging code does not inherently provide an authenticated PNT measurement. Instead, additional signal processing analysis is required, for which Chapter 5 provides a thorough treatment.

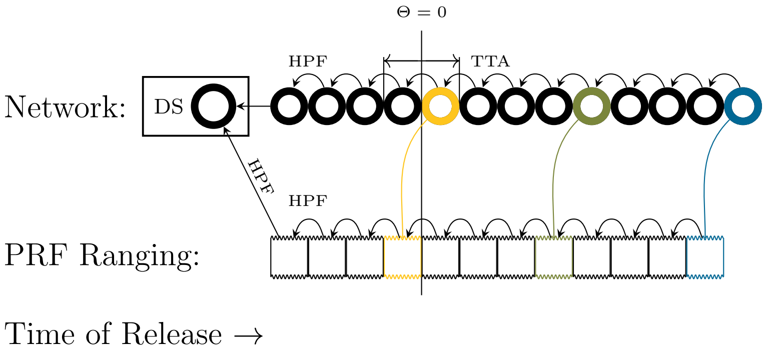


Figure 3.6: Conceptual Diagram Of Geometry With Smallest TTA Possible.

With this theoretical geometry, the output of HPF serves as the ranging code. This geometry cannot be used with multiple satellites or to sign other items. The entire ranging code is distributed after use in the ranging channel. Since the ranging code derives from a PRF, the stream is not compressible. Here, $\Theta = 0$ and the length of the TTA is twice the length of the ranging code distribution. This signal is impractical for numerous reasons, including (1) a perfect $\Theta = 0$ clock, (2) the short output of the HPF serving as a ranging code without any lengthening, and (3) the challenge with distributing the usually high frequency ranging code over a network data channel.

### Practical Computational Concerns

On the question of what is theoretically achievable, Fig. 3.6 depicts a theoretical signal with the shortest TTA. The signal is also impractical because $\Theta = 0$ in the construction. However, Fig. 3.6 depicts the TTA horizon.

In Fig. 3.6, the direct output of HPF is the ranging code, meaning the hash path is not used for anything else. The entire hash path is distributed via a network, and each hash point is released immediately after use in the ranging code. Hence, $\Theta = 0$ and the TTA is twice the distribution length of the output of HPF.

This signal is not practical. As a representative comparison, GPS's C/A ranging code is distributed at 1.023 Mbps. If a ranging signal like that in Fig. 3.6 was to replace the GPS C/A signal, the network connection would need to be 1.023 Mbps for each user. Moreover, GPS C/A signal ranging code is 1023 bits: the smallest currently among GNSS. This means several ***sequential*** hashing operations for a single range measurement in tracking and thousands more for tracking acquisition. KDF can be

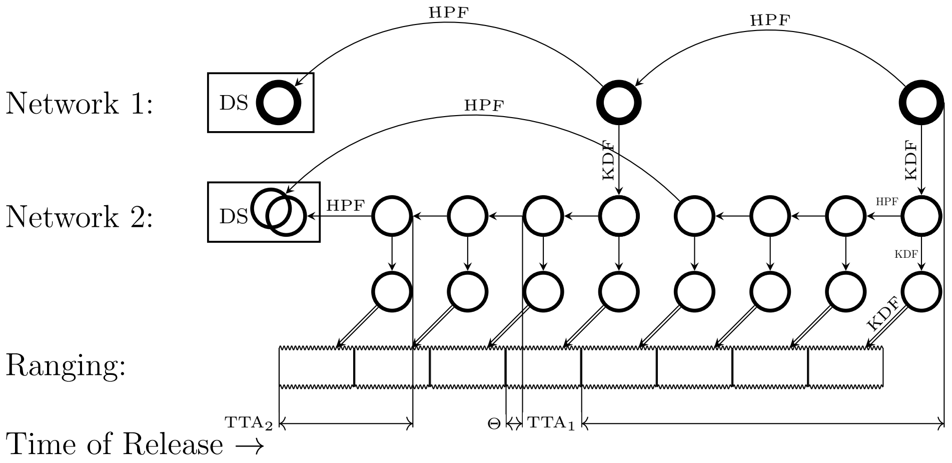


Figure 3.7: Conceptual Diagram Of A Dual Multi-cadence Distribution Construction For A PRF Ranging Code.

In the diagram, the GNSS system provides a PRF ranging code with two distribution cadences. Network 2 has a shorter TTA than Network 1, but Network 1 requires less data transmission to use the signal. Network 2 would provide a premium service compared to Network 1. Regardless of which network is used, the receiver must comply with Θ. HPF is needed when the one-way arrows point against time from hash point to hash point to provide a ***hiding*** commitment. Otherwise, KDF can be used in parallel.

done in ***parallel***, and there is a wider breadth of cryptographic primitive that may have more favorable computational loads. Over an entire ranging code segment, the only time sequential hashing using HPF needed is when a group of receivers receives hash points at that HPF cadence, like in Fig. 3.7.

In a simple authenticated ranging construction (either fully PRF or watermarked), each consecutive hash point will allow a receiver to derive a consecutive authenticated ranging code. In between hash point distributions, the ranging code can be derived in parallel with KDF. The only time HPF is needed in between hash point distributions is with non-diverging and diverging geometries supporting multi-cadence distribution with hiding commitments, like in Fig. 3.7. In the multi-cadence case, KDF can be used directly from the fastest cadence to the authenticated ranging code.

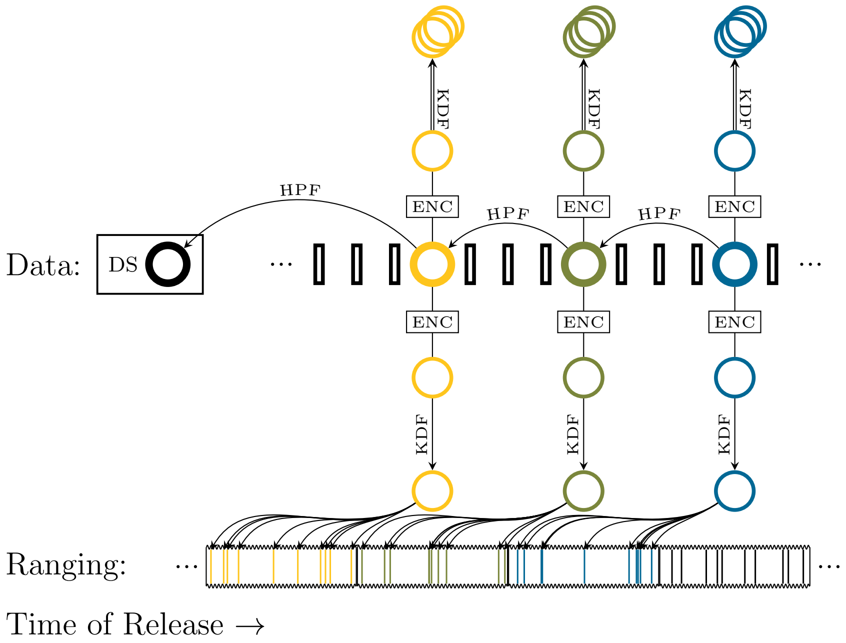


Figure 3.8: Conceptual Diagram Of Incorporating Encryption To Restrict Feature Access To Paying Subscribers Or Authorized Users.

> With this geometry, since the ranging authentication is with watermarking, anyone can track the signal with the known ranging code and observe the distributed hash point. Since there is an encryption operation between each hash point and the KDF operations that derive CMTs and watermarks, only paying subscribers or authorized users can derive those elements. In this case, there is a separate encryption for data and ranging authentication, enabling three tiers of service. The encryption operation is not one way and does not provide any authentication support. Instead, only the delayed-release commitments from TESLA provide authentication.

### 3.1.3 Restricted Access

If the GNSS constellation wishes to restrict access to authentication features (e.g., to generate revenue), the GNSS designer can utilize a geometry like that in Fig. 3.8. To restrict access, the GNSS can insert an encryption operation in between the hash point and the KDF operations using an access key (e.g., with a subscription). In the case of Fig. 3.8, the GNSS provides three levels of service: free data and ranging, authenticated data, and authenticated ranging.

When distributing the access keys, the constellation should separately authenticate the distribution of the access keys to ensure adversaries cannot exploit manipulating them.

To provably prohibit access to non-subscribers, one must prohibit pirate decryption, which is practically impossible. The access restriction is undone if a paying subscriber or authorized user leaks their key. From a practical point of view, the access control only makes utilizing the service without the access key more difficult. From an authentication perspective, the adversary already has the access key, and therefore, the operation does not get an arrow in Fig. 3.8. Since the TESLA schedule is maintained independent of the access key framework and the encryption operation is bijective, the authentication security still holds.

From a receiver computational perspective, it is better to place the encryption operation before the KDF operation, like in Fig. 3.8. If the encryption operation comes after each KDF operation, then the receiver would need to do additional encryption operations (since every parallel KDF would need an encryption operation).

### 3.1.4 Constellation-wide TESLA

Because KDF enables the GNSS to generate additional key material, naturally, a single hash point can authenticate multiple satellites, like in Fig. 3.9. This is the case with Galileo's OSNMA, where the entire constellation utilizes the same TESLA hash point at the same time, and each satellite transmits the same TESLA hash point redundantly [39].

Fig. 3.9 depicts a construction where a single hash point among the message streams of a constellation of GNSS satellites authenticates all information. All information includes the messages in a data channel and the ranging codes. In Fig. 3.9, information (e.g., navigation messages and ranging codes) depicted overlapping indicates uniqueness among the satellite streams; information (e.g., hash points and DSs) not depicted overlapping indicates sameness among the satellites streams. Fig. 3.9 depicts three GNSS satellites simultaneously, but the principle can be extended to any number of satellites. The redundancy of the hash point distribution among each of

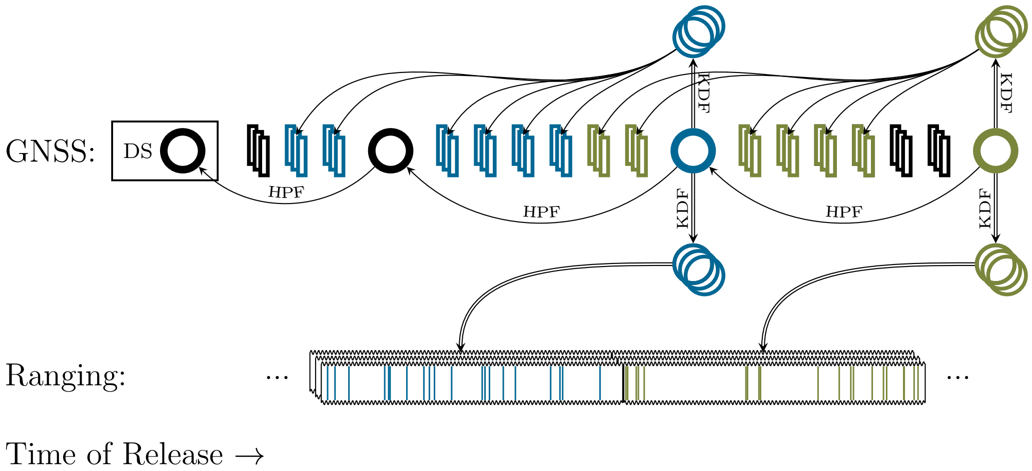


Figure 3.9: Conceptual Diagram Of Constellation-wide TESLA.

In the diagram, a single hash path authenticates all information over the entire constellation: messages, watermarks, and ranging codes. Overlapping items indicate uniqueness among all the message streams, such as messages, watermarks, and ranging codes. Non-overlapping items indicate sameness among all the message streams, such as hash point and DSs of HPEs. The diagram depicts a three-satellite constellation, but the principle can be extended to any number of satellites.

the satellite streams in Fig. 3.9 motivates further hash point distribution optimization discussion in Section 3.2.

## 3.2 Hash Point Distribution

In Section 3.1, I detailed how to use KDF and HPF to generate geometries useful to the GNSS designer. The consequence of utilizing these geometries is shrinking the distributed authentication information by having all authentication information over an interval derived from a single hash point. To derive a PNT measurement, the receiver must measure its range to at least four satellites. Given the multiple GNSS constellations presently available, a receiver will usually range itself to more than four. For a low-earth orbit (LEO) constellation, the number of ranges available is expected to be much larger. In this section, I discuss strategies for efficiently distributing that single hash point in the context of a constellation of GNSS satellites behaving as parallel, redundant channels.

In Fig. 3.9, the hash points are not depicted overlapping since the geometry has the entire constellation utilizing a single hash point over an interval. As depicted, each satellite is distributing the same information simultaneously. Having each of the satellites transmit redundant information would be wasteful. Instead, individual satellites and frequencies should take turns distributing the hash point. When a group is not distributing the hash point, the group can distribute non-authentication information or distribute TESLA maintenance data (see Chapter 4) to better use the available data bandwidth. Hereafter, I discuss three hash point distribution strategies: interleaved, paged, and geometric.

### 3.2.1 Random Interleaved Hash Point Distribution

In an Interleaved Hash Point distribution strategy, the entire constellation utilizes the same hash path. Satellites and frequencies are divided into groups, and the groups randomly take turns distributing the hash point. Fig. 3.10 depicts an Interleaved Hash Point distribution strategy.

Fig. 3.10 utilizes a construction of three groups, although the general observations discussed here apply to a constellation of any number of groups. As illustrated, all groups distribute the DS TESLA maintenance data (see Chapter 4). If an unlucky receiver does not receive the hash point from a particular group, it can derive the hash point from the next distribution from another group.

Suppose that the satellites are randomly assigned groups, and a random group distributes each hash point. In other words, the satellites randomly take turns distributing the hash point without regard to what the other satellites are doing. Suppose there are $G$ groups, and a receiver observes and tracks $V$ satellites in view. For the sake of mathematical brevity, let us assume that $G$ evenly divides the number of satellites and that the constellation is single frequency.

When a hash point is distributed, the chance that a receiver receives the hash point may be modeled as the constellation undergoing a random draw. The probability that a particular satellite does not transmit the hash point is $\frac{G-1}{G}$. Since the random draw

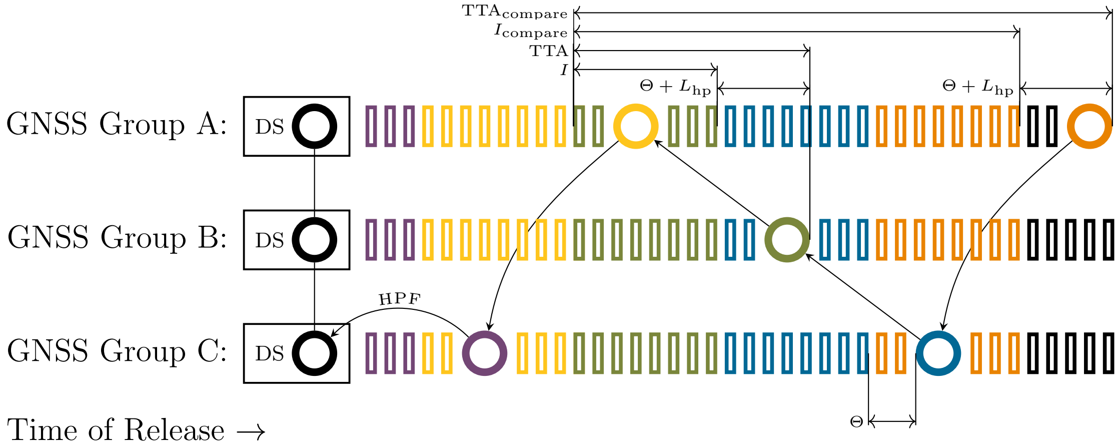


Figure 3.10: Conceptual Diagram Of Interleaved Hash Point Distribution.

A single hash path is used for the entire constellation's message authentication and ranging authentication (not depicted). The constellation is divided into groups (three groups depicted). The groups take turns distributing the hash point. The colors correspond to which hash point authenticates what information. When an unlucky receiver does not observe the hash point from the responsible group, it can derive the missed hash point from the next distribution. The diagram depicts the TTA with a fair comparison with respect to data bandwidth use. The fair comparison is with a receiver only observing one satellite stream as if the other hash points are missed and must be derived from each third distribution. Ignoring message losses, the TTA for the interleaving strategy is approximately the group count times faster, before accounting for Θ and the hash point length.

is repeated for each distribution (and not a shuffle), the events are independent when considering the $V$ in view. Therefore, the probability of receiving the hash point is

$$\Pr(\text{receipt} \mid \text{interleave}, \text{single distribution}) = 1 - \left(\frac{G-1}{G}\right)^{V} . \tag{3.3}$$

Eq. (3.3) does not provide a fair comparison to a non-interleaving strategy with respect to the probability of receipt because of the bandwidth savings. If the hash point is missed, the receiver can derive it from the next hash point distribution, and so on. To account for this effect and provide a fair comparison, the interleaving constellation has $G$ attempts to distribute the hash point within the fair comparison. Since all of these attempts are independent, the fair comparison probability of receipt

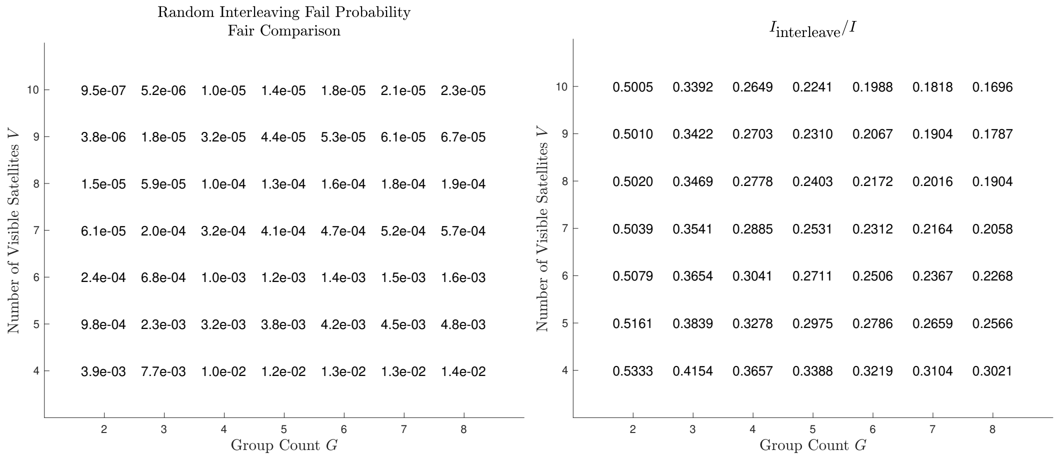


Figure 3.11: Interleaving Fair Comparison Failure Probability And Interval Improvement.

The left figure provides a fair comparison of the probability that an interleaving strategy has a worse TTA. This occurs when each of the $V$ satellites in view does not provide the hash point $G$ consecutive times. The right figure provides the expectation of the interleaving interval size against the naive interval size. Because only a single satellite needs to provide the hash point, the effective interval $I$ (and therefore the TTA) will decrease the overwhelming majority of the time. Since the interval is decreased (rather than devoting extra bandwidth), the TTA decreases in expectation approaching $I/G$ as $V$ increases.

is

$$\Pr(\text{receipt} \mid \text{interleave}, \text{fair comparison}) = 1 - \left(\frac{G-1}{G}\right)^{GV} . \tag{3.4}$$

With $G$ groups, nominally, the new interval is divided by $G$, but there is now a non-zero chance of failed transmission. The number of attempts needed to transmit the hash point follows a geometric distribution. Therefore the expectation for the effective interval, $I_{\text{interleave}}$, is

$$\mathbb{E}\left[I_{\text{interleave}}\right] = \frac{I}{G}\frac{1}{\Pr(\text{receipt} \mid \text{interleave}, \text{single distribution})} \tag{3.5}$$

$$\mathbb{E}\left[I_{\text{interleave}}\right] = \frac{I}{G}\frac{1}{1-\left(\frac{G-1}{G}\right)^{V}} . \tag{3.6}$$

Fig. 3.11 provides two comparison diagrams that demonstrate the TTA advantage with using interleaving hash point distribution strategy. The left provides the fair comparison transmission failure probability (i.e., when the interleaving strategy increases the TTA over a non-interleaving strategy). This occurs when each of the $V$ satellites in view does not provide the hash point $G$ consecutive times. The right provides the new interval in expectation, which approaches $\frac{I}{G}$ as $V$ increases.

This interleaving strategy is easy to model to determine the performance gains. However, a small performance improvement can be obtained by implementing a shuffle strategy [8]. For instance, rather than having each satellite draw to distribute the hash point without considering the other satellites, the satellites could be randomly ***shuffled*** into $G$ groups evenly. That is, the random shuffle guarantees that the satellites are shuffled into identically sized groups, removing the chance that a particular uneven group drawing unfavorably affects hash point transmission.

Current GNSS generally have rigid message schedules in the data stream. Often, the authentication information is placed in spare bits that require the information to be split among many tiny pages. This presents a challenge to random message assignments in real-time, for which there are mitigations. For instance, suppose the entire constellation must broadcast authentication information at the same time. Then the constellation can devote non-hash-point-broadcasting satellites to TESLA maintenance information (see Chapter 4). In fact, the GNSS designer can elect for any distribution among the hash point and TESLA maintenance information to meet its TTA and cold start requirements.

Moreover, transmitting random assignments need not require additional bandwidth to provide page identities (for the receiver to associate pages). The TESLA hash path and the DS can provide the seed for a pseudorandom number generator. For instance, a constellation can derive all the randomness it needs using KDF on a previously-released hash point or the hash of the current TESLA DS. This means the constellation and receiver can determine all page identities deterministically from available information.

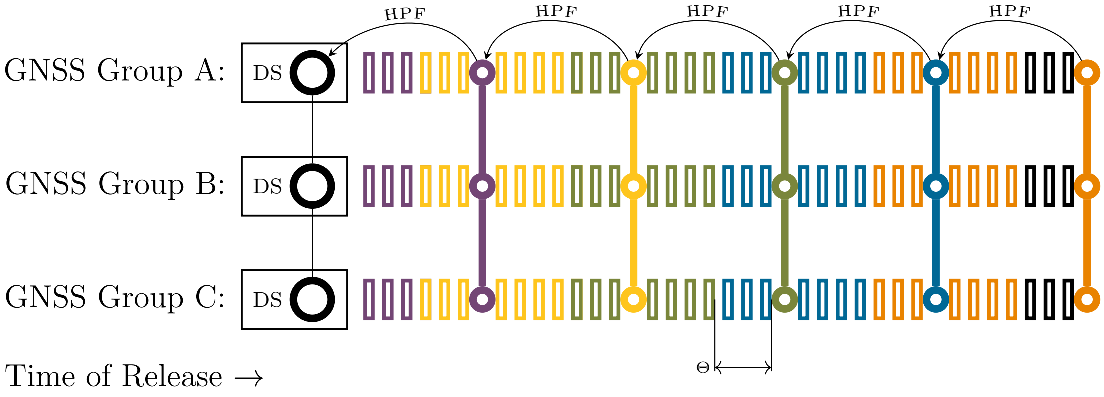


Figure 3.12: Conceptual Diagram Of Paged Hash Point Distribution.

A single hash path is used for the entire constellation's message authentication and ranging authentication (not depicted). The constellation is divided into groups (three groups depicted). The hash point is divided into pages, with each group distributing a different page simultaneously. The colors correspond to which hash point authenticates what information. When an unlock receiver does not observe a page, it can derive the missed hash point from the next distribution.

### 3.2.2 Random Paged Hash Point Distribution

Current GNSS generally have rigid message schedules in the data stream. Suppose the constellation must devote bits to distribute hash points among each satellite simultaneously. And, unlike Section 3.2.1, the constellation cannot fill non-hash-point-broadcasting satellites with other bandwidth. Then the constellation can split the hash point into paged sections among the groups for simultaneous transmission among the constellation for almost the same TTA gain as that of the interleaving strategy from Section 3.2.1. For this strategy, all groups should transmit the paged sections simultaneously. Otherwise, the length of time needed to transmit a hash point ($L_{\text{hp}}$) effectively increases, causing the TTA to increase. Fig. 3.12 provides a conceptual diagram of this hash point distribution strategy.

As with the interleaving case with Section 3.2.1, I will only model the random drawing case. However, for the same reasons described in Section 3.2.1, a small performance improvement can be obtained when using a shuffle [8]. And, for mathematical brevity, I assume that $G$ evenly divides the number of satellites.

A receiver receives the distribution only if each unique page is received. To compute the probability of receipt, I use the inclusion-exclusion principle to derive

$$\Pr(\text{receipt} \mid \text{paged, single distribution}) = \begin{cases} 0 & \text{if } G > V \\ 1 + \sum_{i=1}^{G-1} (-1)^i \cdot \binom{G}{G-i} \cdot \left(\frac{G-i}{G}\right)^V & \text{otherwise} . \end{cases} \tag{3.7}$$

A partial distribution at one time does not assist with recovery in later distributions. Therefore, the fair comparison probability is

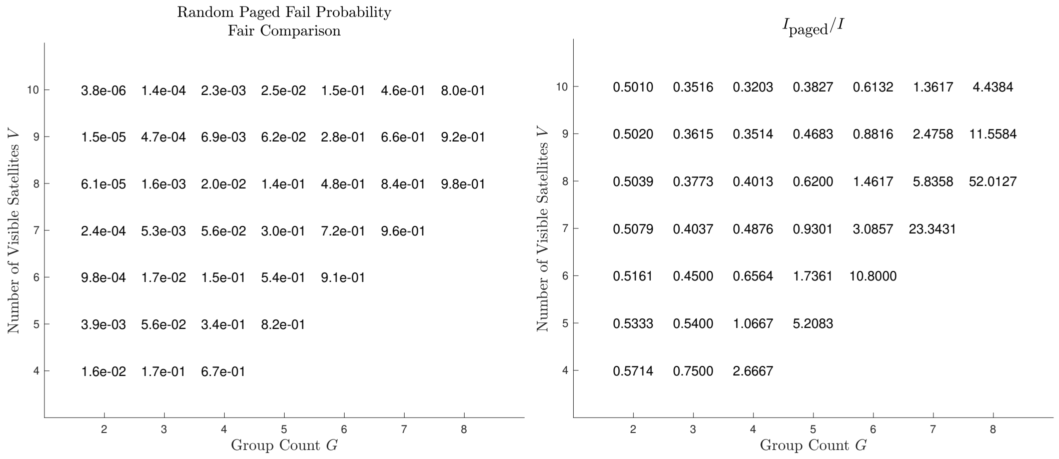
$$\Pr(\text{receipt} \mid \text{paged}, \text{fair comparison}) = 1 - (1 - \Pr(\text{receipt} \mid \text{paged, single distribution}))^G . \tag{3.8}$$

As with Section 3.2.1, the number of attempts needed to make a distribution follows the geometric distribution. Therefore,

$$\mathbb{E}\left[I_{\text{paged}}\right] = \frac{I}{G} \frac{1}{\Pr(\text{receipt} \mid \text{paged}, \text{single distribution})} . \tag{3.9}$$

Fig. 3.13 provides two comparison diagrams that demonstrate the advantage of using paged hash point distribution strategy. The left provides the fair comparison transmission failure probability when the paging strategy increases the TTA over a non-paging strategy. This occurs when each of the $V$ satellites in view does not provide the hash point $G$ consecutive times. The right provides the new interval in expectation, which approaches $\frac{I}{G}$ as $V$ increases.

The paged strategy is a simple design improvement. The designer takes the non-paged bandwidth allotment and divides by $G$. However, the performance of the paged improvement is visibly worse than the interleaved strategy of Fig. 3.11, and there are system designs that perform worse than the non-paged strategy when $V$ is not high enough.



Random Paged Fail Probability
Fair Comparison

| $V$ \ $G$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 10 | 3.8e-06 | 1.4e-04 | 2.3e-03 | 2.5e-02 | 1.5e-01 | 4.6e-01 | 8.0e-01 |
| 9 | 1.5e-05 | 4.7e-04 | 6.9e-03 | 6.2e-02 | 2.8e-01 | 6.6e-01 | 9.2e-01 |
| 8 | 6.1e-05 | 1.6e-03 | 2.0e-02 | 1.4e-01 | 4.8e-01 | 8.4e-01 | 9.8e-01 |
| 7 | 2.4e-04 | 5.3e-03 | 5.6e-02 | 3.0e-01 | 7.2e-01 | 9.6e-01 | |
| 6 | 9.8e-04 | 1.7e-02 | 1.5e-01 | 5.4e-01 | 9.1e-01 | | |
| 5 | 3.9e-03 | 5.6e-02 | 3.4e-01 | 8.2e-01 | | | |
| 4 | 1.6e-02 | 1.7e-01 | 6.7e-01 | | | | |

$I_{\text{paged}}/I$

| $V$ \ $G$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 10 | 0.5010 | 0.3516 | 0.3203 | 0.3827 | 0.6132 | 1.3617 | 4.4384 |
| 9 | 0.5020 | 0.3615 | 0.3514 | 0.4683 | 0.8816 | 2.4758 | 11.5584 |
| 8 | 0.5039 | 0.3773 | 0.4013 | 0.6200 | 1.4617 | 5.8358 | 52.0127 |
| 7 | 0.5079 | 0.4037 | 0.4876 | 0.9301 | 3.0857 | 23.3431 | |
| 6 | 0.5161 | 0.4500 | 0.6564 | 1.7361 | 10.8000 | | |
| 5 | 0.5333 | 0.5400 | 1.0667 | 5.2083 | | | |
| 4 | 0.5714 | 0.7500 | 2.6667 | | | | |

Figure 3.13: Paged Fair Comparison Failure Probability And Interval Improvement.

> The left figure provides a fair comparison of the probability that an interleaving strategy has a worse TTA. This occurs when each of the $V$ satellites in view does not provide the hash point $G$ consecutive times. The right figure provides the expectation of the interleaving interval size against the naive interval size. Because only a single satellite needs to provide the hash point, this will occur most of the time. Since the interval is decreased (rather than devoting extra bandwidth), the TTA decreases in expectation approaching $I/G$ as $V$ increases.

### 3.2.3 Geometric Hash Point Distribution

In Sections 3.2.1 and 3.2.2, the randomness provides an easy path to modeling performance improvement by removing the orbital geometry dependence. They trade knowledge of the constellation for highly probable performance improvements. Yet, there should be a way to exploit the constellation's geometry to provide a hash point distribution strategy that will provide similar performance improvement with high probability.

GPS satellites are approximately divided into six orbital planes. For a receiver to deduce a useful PNT solution, the solution needs to derive from ranges from a diverse satellite geometry. For instance, if a PNT solution derives from satellites from only the same orbital plane, then the PNT will have less or no precision. GNSS constellations are designed to limit this scenario, so the hash point distribution can piggyback off of this design effort.

A simple way to leverage this for hash point distribution for GPS is to divide the constellation into groups by GPS orbit ascending node. Among the GPS constellation are six orbital planes with orbit ascending nodes separated by 60 degrees. One could divide the satellites into three groups with opposite ascending nodes (e.g., $0°/180°, 60°/240°, 120°/300°$). When a receiver observes at least three orbital planes, then it will receive the hash point for every distribution. If a receiver does not, then it will not receive every distribution at the fastest possible cadence, but it would likely receive it at the following distribution since the groups would take turns distributing the hash point. And in that scenario, the PNT solution would likely not be as good anyway. Simulation would be needed to verify the efficacy of this distribution strategy.

## 3.3 Example Signal Sketches

Whereas Sections 3.1 and 3.2 discussed design principles for applying TESLA to GNSS, this section uses those principles to suggest new example signal designs. Section 3.3.1 makes several suggestions to the Chimera signal to decrease TTA. Section 3.3.2 suggests incorporating SBAS to decrease TTA within SBAS service volumes principally for aviation users. Section 3.3.3 suggests modifying the key management of modern encrypted signals to provide pseudoauthentication for open users. Section 3.3.4 suggests a built-from-scratch dual-channel authenticated GNSS service. And Section 3.3.5 suggests a network-only signal and discusses what can be achieved.

### 3.3.1 Multi-Cadence Chimera with Random Interleaving

The principles from this chapter can help Chimera achieve more favorable performance. In this section, I apply the random interleaved hash point distribution strategy to Chimera. This, with the switch from ECDSA-only to TESLA-ECDSA, enables about a six-fold TTA performance improvement. Moreover, I apply diverging geometries to allow for the two Chimera-proposed watermarks to be combined, halving the required signal degradation. In Chapter 5, I further decrease the required watermark

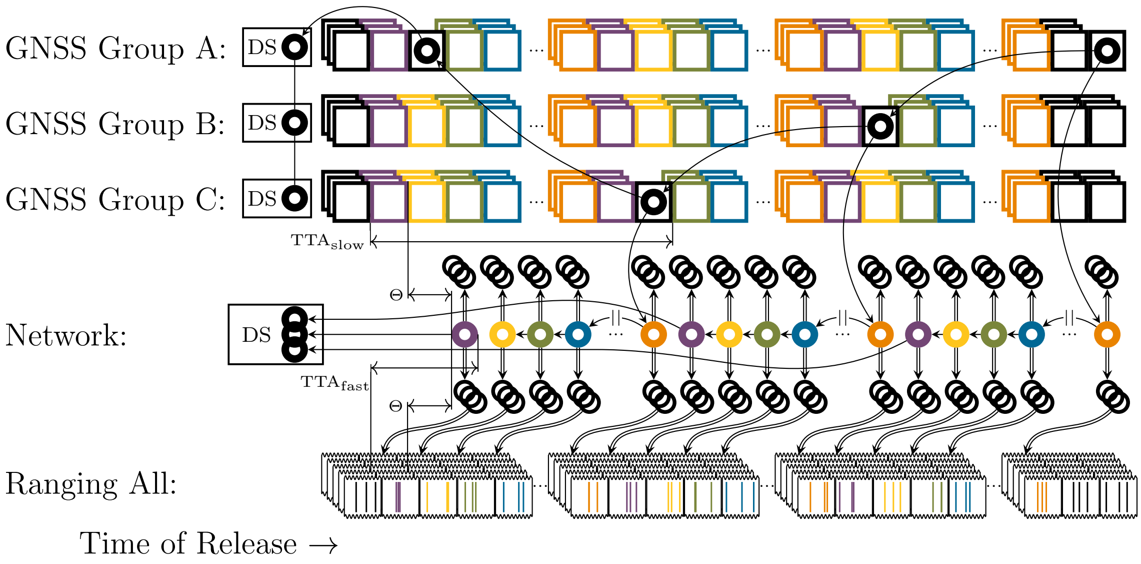


Figure 3.14: Conceptual Diagram of Chimera Combined Slow And Fast Channel Watermark With Interleaving Satellite Hash Point Distribution.

In this Chimera design, the entire constellation is authenticated with a single hash path: navigation message and all watermarks. The hash point is distributed using the random interleaved distribution strategy from Section 3.2.1. And the constellation exploits diverging KDF geometry from Section 3.1.1 to allow a single watermark for network and non-network users. The principle advantage is that the ranging code need not be degraded twice; the principle disadvantage is that the clock synchronization requirement becomes the same for all users. The colors correspond to what hash point derives the CMT of each message or derives the watermark within the ranging code. I note that the main hash path and each of the diverging paths are the same for all satellites (and not depicted overlapping), until KDF derives CMT keys and watermark seeds for each satellite. The constellation is divided into three groups, and when a group broadcasts a hash point, all satellites in that group broadcast the same hash point data to support the transmission efficacy from Section 3.2.1.

degradation. Fig. 3.14 provides the TESLA sketch diagram on how to achieve these performance improvements.

### Constellation-Wide Hash Path with Random Interleaving

The Chimera navigation stream is composed of 18-second messages with three subframes. The first subframe helps with receiver time synchronization. The second subframe includes clock and ephemeris data. The third subframe has 250 bits of variable or unassigned data, which includes the proposed authentication information. In Fig. 3.14, within the GNSS groups, each rectangle represents a message. If the rectangle contains a circle, then ***subframe 3*** would carry the hash point.

With the entire constellation, the satellites are assigned into three groups. The group assignments could be based on an orbital plane or randomly. For each TESLA interval, a random group will distribute the hash point within its subframe 3. When the subframe three is not distributing a hash point, it will distribute other Chimera information, such as the CMTs and ECDSA information (see Chapter 4). Hence, within Fig. 3.14, when a group is not transmitting the hash point, it is presented with overlapping rectangles indicating different messages among the satellites in the group. When a group is transmitting a hash point, all the satellites within the group are transmitting the hash point, so there are no overlapping rectangles.

At present, the proposed TTA is approximately 3 minutes. The use of TESLA and random interleaved hash point distribution presents bandwidth savings. Rather than allocating the bandwidth saved for other purposes, I propose decreasing the TTA. Switching from ECDSA-256 signatures to 128-bit TESLA with 32-bit CMT decreases each authentication from 512 bits to 160 bits. Moreover, by having the constellation share a hash path and randomly distribute them, the 128-bit TESLA hash point distribution effectively goes down by a third, meaning an effective bit count of 75. The CMT does not decrease by a third because each satellite needs its own CMT since the receiver will only observe a subset of satellites. While this savings purports about a 6.8x performance improvement on bandwidth, this analysis neglects several important factors such as the TESLA commitment reveal delay ($\Theta$), $L_{\text{hp}}$, the DS maintenance, and signing multiple messages per TESLA interval. For instance, Chimera could sign each message separately with a 32-bit CMT rather than the entire interval with a single CMT for loss tolerance. Moreover, there is a non-zero probability of not receiving the hash-point distributing group; however, from Section 3.2.1, the probabilities and TESLA loss tolerance property favorably helps. Therefore, I estimate a six-fold TTA performance improvement (i.e., a TTA of 30 seconds) is reasonably achievable using these methods.

### Multi-Cadence, Diverging Geometry, and a Single Watermark

Chimera proposes two independent watermarks. One is distributed via the GNSS data channel and the other via network. The network watermark has a much faster cadence, enabling network-connected receivers to have a much faster ranging TTA. Because these watermarks are independent, there are two degradations on the signal.

Using diverging geometries from Fig. 3.3, the two watermarks can be combined. In Fig. 3.14, each of the navigation message keys and watermark seeds derives from a diverging path off of the original hash path. The diverging path is concurrent for all satellites to save with receiver computation and the required data distributed over the network. The network TTA shrinks to meet the TTA performance requirement without affecting the non-network TTA. However, the principal advantage is that the amount of watermark degradation halves. Chapter 5 discusses how the degradation can be further improved.

This advantage does not come for free. The principal disadvantage is that the $\Theta$ is now the same for both network and non-network user groups. This makes non-network receivers have a tighter required synchronization, increasing the expense of the onboard GNSS-independent clock (GIC) for non-network receivers.

Combining the watermark but requiring better clocks on non-network receivers is a favorable trade. Overall, having network-capable vehicles with GNSS makes those vehicles cheaper and have better performance (e.g., for non-navigation reasons). That is, incorporating the internet makes things easier, not harder. For instance, a network receiver can more frequently correct its GIC, decreasing the GIC drift requirements. The safety-of-life user base that insists on having a network-independent signal is principally civilian and military aviation, where cockpit security concerns and challenges with incorporating a cockpit network connection prohibit incorporating a network. Now, compare to another safety-of-life user base with easy network access, such as autonomous cars and drones.

The non-network user base (i.e., aviation) can tolerate more expensive GNSS receivers better than the network user base (i.e., cars and drones). Because of the relative expense in these two user groups, the expense incurred by requiring non-network receivers to have a better GIC would not be a substantial concern for the non-network user base. The cost associated with updating all existing receivers to handle twice the needed degradation (or changing the satellite signal power) makes this trade favorable. Therefore, trading a less-degrading watermark for a more expensive non-network receiver GICs presents a trade worth making.

### 3.3.2 Multi-Cadence with SBAS

Within GNSS, if the core constellation and SBAS satellites coordinate and share a TESLA hash path, then a non-diverging geometry can improve TTA performance when a receiver is within an SBAS service volume. This would be particularly useful for aviation, where receivers do not have a network connection and TTA requirements can vary among flight stages [79]. Takeoff and landing flight stages have a higher risk profile than open flight for which SBAS is already well suited to assist. Suppose regulatory agencies demand that the TTA be tighter than what is achievable with core GNSS during only the higher-risk takeoff and landing flight stages. Given SBAS current regional coverage over terminal control areas, a coordinated core and SBAS authentication scheme would work well to meet tighter TTA requirements. Fig. 3.15 provides an example of non-diverging construction for this purpose.

The design in Fig. 3.15 offers three TTA cadences: network, SBAS, and core. Network users have the best TTA, followed by 6-11 seconds for SBAS users [13], and then core-only users. If an aviation user utilizes SBAS, then they should only operate on the slowest TTA when operating in between service volumes where open flight poses the smallest risk. The core and network cadences follow from Section 3.3.1; therefore, the SBAS is a simple non-diverging geometry extension on the design proposed by Section 3.3.1. For conceptual diagram simplicity, Fig. 3.15 does not depict the random interleaving hash point distribution among the core constellation. Because current SBAS requirements do not require aviation users to track each of the SBAS satellites [87] and the realities of the geostationary orbit geometry, it is not appropriate to employ a hash point distribution strategy from Section 3.2 on the SBAS

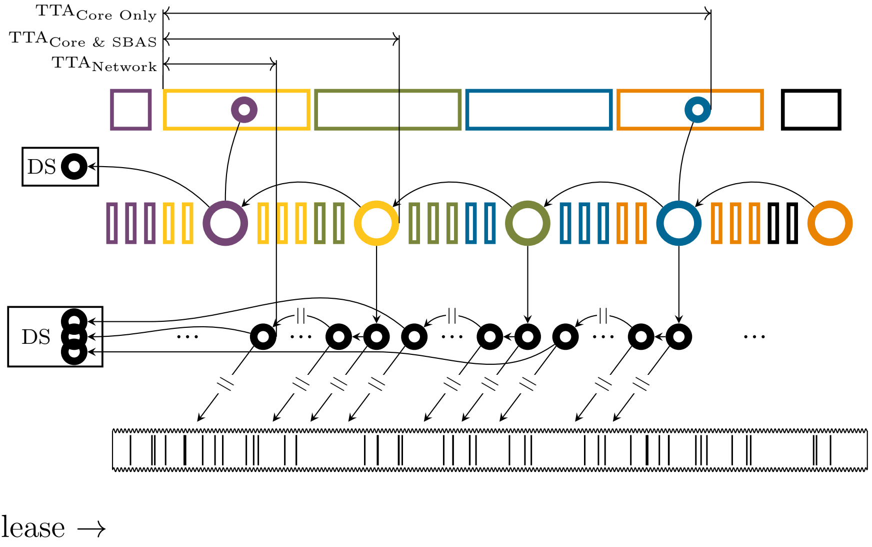

$\text{TTA}_{\text{Core Only}}$
$\text{TTA}_{\text{Core \& SBAS}}$
$\text{TTA}_{\text{Network}}$
Core Data:
DS
SBAS:
Network:
DS
Ranging:
Time of Release →

Figure 3.15: Conceptual Diagram With Non-Diverging Geometry With SBAS And Network Multi-cadence Distribution.

Core GNSS and SBAS share a hash path. In addition to meeting its six-second authenticated integrity requirement from [13], SBAS, authenticates core satellites visible over its service volume. The design uses non-diverging geometry with SBAS and diverging geometry for watermarking (following from Section 3.3.1). Not all optimizations are depicted, such as random interleaving for the core constellation hash point distribution and constellation-wide path sharing (e.g., overlapping rectangles). With this system, the network users have the best TTA, aviation users have a 6-11 second TTA when utilizing SBAS during takeoff and landing, and others have the longer TTA (e.g., during unobstructed open flight outside SBAS service volumes).

distribution channel. Moreover, Fig. 3.15 does not depict the constellation-wide hash path sharing optimizations.

Using two layers of diverging geometry like Fig. 3.3 so that the SBAS hash points diverge from the core satellite hash points would not work. Doing so would require that SBAS distribute a DS frequently, which is not feasible under the data bandwidth limitation.

Given the European GNSS constellations, Galileo and European Geostationary Navigation Overlay Service (EGNOS), are designed and managed by comparatively close personnel (when compared to their American counterparts), there is an opportunity for a quicker adoption of this strategy in the context of SBAS TESLA standardization. However, the present Galileo interface control document (ICD) does not parameterize the number of hashes in between Galileo hash points, meaning this strategy already presents a backward compatibility issue. But such a change would not require making any changes to the message structure, requiring only document and software updates to instruct receivers to hash multiple times in between OSNMA hash point distributions.

### 3.3.3 Pseudoauthentication With Encrypted Signals

Modern encrypted signals could provide pseudoauthentication. Rather than using independent encryption keys for encrypted signals, the encryption keys can derive from a hash path. Defense, utility, and other restricted-use stakeholders would retain secure real-time encrypted use by receiving the hash path start to them as privileged users. After a specific key is used, that key is widely broadcast, like any other TESLA scheme. Given the hiding commitment security of the hash path as it is released backward, released hash points do not compromise ***privileged access*** security for the privileged users (until the delayed release). Therefore, this modification would allow open users a delayed cryptographically authenticated signal while retaining a confidential real-time service for privileged users. For this service to be useful, the encryption key change frequency must be fast since the encryption key change frequency determines the time to authentication, like with the signals from the above sections. Fig. 3.16 provides a conceptual construction.

Fig. 3.16 violates the geometric rules from Section 3.1 and does not provide real authentication security. Anyone with the privileged data, or if the privileged data is ever leaked and obtained by a spoofer, can generate false ranging signals that would break authentication security. However, since there are existing deterrents to the leakage of encrypted signal access keys, this system does make spoofing more difficult.

Implementing this design concept could prove easier than other authentication

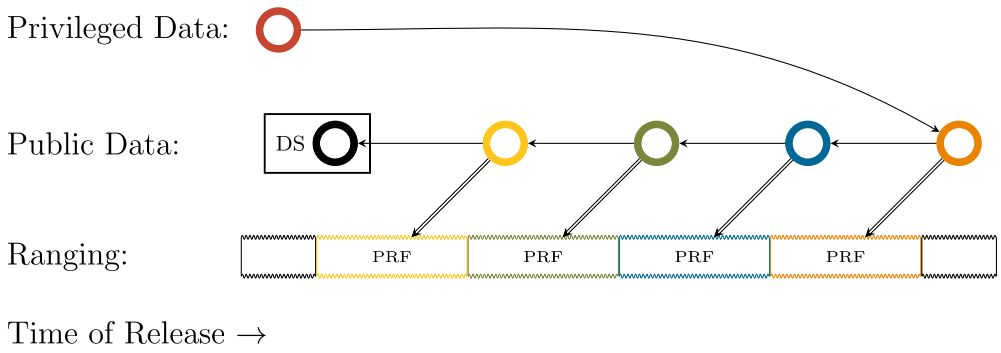


Figure 3.16: Conceptual Diagram Of A Pseudoauthentication With Delayed-public-release Of Encryption Keys Of An Encrypted Signal.

In this design, an encrypted GNSS system provides privileged users real-time access to ranging. As shown in the diagram, the encryption keys derive from a hash path and privileged users receive the entire hash path via the hash path start (from which all other information is derivable). Non-privileged users receive the encrypted keys in reverse-derivation order via another stream (e.g., a network) and perform ranging pseudoauthentication at a delay. The geometry violates the condition from Section 3.1, meaning authentication is not actually provided, but this pseudoauthentication strategy does make spoofing more difficult. Any privileged user, or anyone who is can obtain the privileged data, can break the authentication argument. In this diagram, the intermediate KDF operations and concurrent use of a hash path among satellites is not depicted.

concepts since it is achievable without modifying present signals. Converting encrypted signals into authentication signals is as simple as modifying the key management. Moreover, converting a pseudoauthentication signal to an authentication signal would only require not distributing the keys before the encrypted ranging code transmission (i.e., have no privileged users).

### 3.3.4 Dual-Channel Design

In this section, I explore a GNSS PRF-ranging signal design that requires no external connection. Without an external connection, the signal must have a separate acquisition signal from which the other signals derive, like the dual-channel-in-quadrature GPS C/A and P(Y) signal. Fig. 3.17 provides a conceptual diagram of a suggested signal, where two data and ranging signals are modulated together.

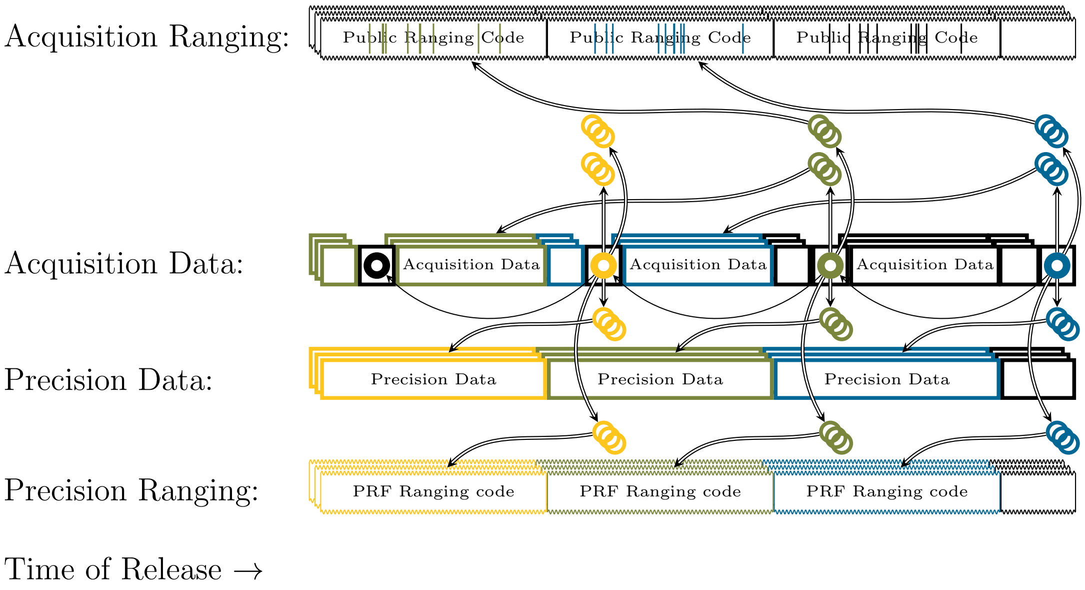


Figure 3.17: Conceptual Diagram Of A Dual Channel Authenticated GNSS Signal.

The receiver uses a public acquisition ranging code (e.g., a Gold Code) to acquire and track an acquisition signal. From the data on the acquisition signal, the receiver recovers a hash point that enables the receiver to track the authenticated PRF ranging signal and receive the data on the precision signal. To support receivers with worse clocks, there is a watermark in the acquisition ranging signal with a looser $\Theta$, and the data within the acquisition signal is also authenticated at a looser $\Theta$. Hence, the colors in the acquisition signal do not align with the precision signal. To shrink the TTA, the acquisition signal should contain minimal amounts of non-hash-point data (and move other data to the precision data signal) so that as much data bandwidth as possible can be devoted to hash point distribution to increase the hash point cadence.

In Fig. 3.17, the receiver uses a public acquisition ranging code (e.g., a Gold Code) to acquire and track the signal. From the data component of the acquisition signal, the receiver recovers each hash point. From each hash point, the receiver derives all of the PRF ranging codes of the entire constellation to compute authenticated ranges. In Fig. 3.17, there are two data channels: acquisition and precision. The design should move as much data from the acquisition data to the precision data as possible so that the hash point cadence can be as fast as possible, decreasing TTA. Ideally, the acquisition data would only contain hash points.

In support of receivers with a worse GIC than precision users, Fig. 3.17 includes a watermark within the acquisition ranging code with a looser $\Theta$. Moreover, the data in the acquisition channel is also authenticated at this slower $\Theta$. Therefore, Fig. 3.17 supports authentication at two different $\Theta$ requirements, and the colors of the acquisition and precision signals do not align.

Several performance optimizations could be implemented that are not depicted. Since the receiver will need to perform the acquisition operations on each of the signals anyway, the constellation could also include the optimizations of Section 3.2 to decrease TTA (by increasing the cold start authentication time). In the case of LEO constellations, the constellation could elect only to transmit the acquisition signal on a subset of satellites and devote more power to the precision signals on the remaining satellites. The constellation must be careful that at least one signal transmitting the acquisition signal is in view throughout the Earth.

### 3.3.5 Network Service Only

GNSS is a good ranging system but a terrible data communication system. On a theme similar to the separation of Chapters 3 and 4, the fastest possible TTA practically achievable is with a network-only service. With a network-only service, GNSS only provides a ranging signal (i.e., without any navigation data). Fig. 3.18 provides a conceptual diagram of a network-only GNSS ranging service.

Modern GNSS devices with access to a network already do not use the navigation data channel on GNSS. This is the entire basis for AGPS in modern smartphones [36]. So then, for a GNSS constellation serving this user group (including autonomous cars), such a service would provide the best possible TTA practical without any burden to distribute data on top of the ranging signal.

## 3.4 Application to SBAS

In this section, I review the strategies of this chapter applied to SBAS in [13]. In Section 4.4, I review the strategies of Chapter 4 applied to SBAS in [13].

A common theme to SBAS authentication is that the scheme need only prohibit

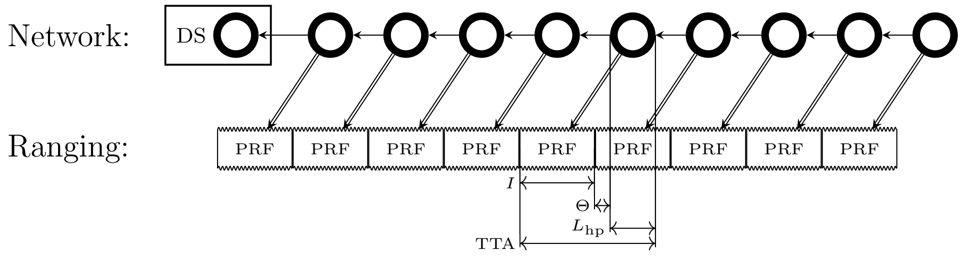


Figure 3.18: Conceptual Diagram With A Network-only Authenticated Ranging Signal.

The receiver uses a network to receive the hash path from which PRF ranging codes are generated. Given the challenges with receiving data over low-bandwidth GNSS signals, and given that many users utilize networks to receive the navigation data contained (e.g., AGPS [36]), this signal would provide a more favorable TTA to network GNSS users.

a receiver from utilizing forged information. This means that if a message were broadcast telling a receiver not to use SBAS, it need not be strictly authenticated (e.g., a fact leveraged in Fig. 3.20). While doing so lends an easier pathway for a spoofer to deny a receiver service, in the context of civil aviation, denial of service is ok because there is a pilot.

The SBAS standard provides for two channels: in-phase and quadrature-phase; however, only the in-phase channel is presently used. To incorporate authentication to SBAS, it would be very tempting to utilize the quadrature-phase channel, and others have investigated it [47, 73]. Were SBAS constellations to spontaneously begin transmitting over the quadrature channel, the present satellites would have to divert power away from the current in-phase channel. Decreasing the power would adversely affect receivers operating at SBAS service volume boundaries, rendering such a scheme unacceptable. Luckily, there is enough data bandwidth on the SBAS in-phase channel to incorporate a TESLA scheme that meets design requirements.

In [70], they suggested appending a single message type (MT) to the SBAS schedule: MT50 for L5 and MT20 for L1. To ensure backward compatibility, rather than having each SBAS message deliver its own keyed-hash message authentication code (HMAC), MT50/20 would deliver message HMACs from the previous messages separately. [70] suggests that SBAS sends this MT every six messages and that this MT delivers 190-bits of TESLA authentication data and 26-bits for the TESLA scheme's maintenance. While the scheme requires a steep authentication-message delivery frequency, it does not overburden the SBAS MT schedule. In [13], I remove the 26-bit scheme maintenance information into its own message MT51, as discussed in Section 4.4. The rest of this section discusses the design and implications of MT50 in the improved design from [13].

At the time of writing, SHA256 and HMAC-SHA256 are the ubiquitous and defacto standard for hashing and symmetric message authentication codes. Using what everyone else uses aids in the security of the scheme. If SHA256 were ever broken, it would be widely publicized and quickly deprecated for another well-studied hash function. Moreover, using SHA256 for hashing and HMAC simplifies implementations, making updates and changes easier. In the following sections, the hash function within the HPF is SHA256, and HMAC-SHA256 is utilized for both the KDF and the commitment-MAC function (CMF).

In Section 3.4.1, I discuss hash point and HMAC size for SBAS. In Section 3.4.2, I discuss the construction of MT50 and hash point delivery for SBAS. In Section 3.4.3, I discuss accommodating SBAS alerts with TESLA. In Section 3.4.4, I propose utilizing SBAS for core GNSS constellation navigation message authentication (NMA).

### 3.4.1 Hash Point and HMAC Size

Per [79], the acceptable landing integrity risk for landing operations must be less than $10^{-7}$ per approach. An approach takes 150 seconds, meaning that there are 150 attempts to forge a message over an approach. However, some information available to an SBAS receiver for an approach is received up to 10 minutes earlier. By requiring that the receiver stop authentication for at least one minute after a failed HMAC authentication, the adversary has only 12 attempts to forge a message during an approach [108]. From there, a message spoofing probability of $12 \cdot 2^{-28} < 10^{-7}$ was deemed sufficient for the security requirement per approach.

For the hash point size, [13] selected the standard 128-bit security level. Others have done analysis that suggests that a smaller size could be sufficient for GNSS [32, 70]. This decision to move to 128 was done to ensure the hash path as a whole conforms with a standard security level. Moreover, this decision completely ignores quantum resistance which is not considered feasible for SBAS authentication in general.

Since the hash point is not known to an adversary when the HMACs are released, the probability that an adversary can forge a $b$-bit HMAC is $2^{-b}$. Given the SBAS data bandwidth constraints, really short HMACs were considered. To provide a $10^{-7}$ protection level with 16-bit HMACs, [13] suggested several approaches, including one based on message authentication code (MAC) accumulation [38] (see Section 1.3.2). Two 16-bit HMACs from [13] are sufficient in this case. Given the repetitive nature of data transmission over SBAS, the receiver could insist that certain information be signed by two HMACs. Another suggestion is that if any HMACs were to fail, the receiver would throw out all of the navigation data and restart accumulating SBAS data.

With 16-bit HMACs, five can fit per MT50, and the five provide better loss tolerance properties than one large HMAC, as explained in [70]. However, more recent proposals utilize more advanced error correction techniques that enable larger HMACs, as discussed in Section 3.4.2.

### 3.4.2 MT50: Hash Point and HMAC Delivery

Currently, each SBAS message is accompanied by a cyclic redundancy check (CRC). If SBAS authentication were being designed from scratch, then each message could be accompanied by an HMAC. But to maintain backward compatibility, SBAS message HMACs must be sent in a separate message: MT50. Receivers ignoring authentication can simply ignore MT50s without affecting their current operations. By separating the TESLA maintenance distribution with MT51 (see Section 4.4.1), there is enough space for $b = 128$ bit hash points and HMACs to authenticate five messages. And there is enough room in the SBAS schedule to send an MT50 every sixth message.

However, this MT50 backward compatibility design requirement introduces additional message loss tolerance challenges. When HMACs accompany their messages, then the loss of one message/HMAC does not affect other messages. Because MT50 will deliver groups HMACs, the loss of an MT50 affects the authentication of at least five messages. A single HMAC per MT50 covering multiple messages is not a good idea because then the loss of any one of the five messages would prohibit authentication of any of them (because the receiver must use each received message to generate that HMAC for comparison under TESLA). Therefore, [13, 70] suggested having each MT50 contain five 16-bit HMACs. Then, the loss of a single message does not affect the other messages' authentication (except when the MT50 is lost).

Since the proposal of [13], stakeholders have expressed their concerns with the short 16-bit HMACs and their preference not to use MAC accumulation [38]. MAC accumulation specifies that a message receives multiple authentications before its use (e.g., requiring a message be signed with two separate 16-bit HMACs would provide 32-bit security). MAC accumulation delays TTA and does not alleviate general concerns that 16-bit HMACs are just too short.

Recently, [108] proposed utilizing an advanced error correction scheme enabling larger HMACs between 28 and 36 bits. Let the security level and the bit length of each HMAC be $b_{\text{AMAC}}$. In [108], each message has its own $b_{\text{AMAC}}$-bit HMAC, and then these five HMACs are concatenated and then signed again by another $b_{\text{AMAC}}$-bit HMAC called an AMAC. In addition to the hash point, MT50 will provide this AMAC and two additional $b_{\text{AMAC}}$-bit fields that enable erasure recovery of two out of the five message HMACs. The following paragraphs and Table 3.2 provide a proposed definition of MT50 with the bit allocations for $b_{\text{AMAC}} = 36$ for L5 (see [108] for the other versions, including those compatible with L1).

Each message $m_j$, sent at $t_j$, will get its own HMAC key $k_j$, complying with

| Preamble | MT | ER1 | ER2 | AMAC | Hash Point | Spare |
|---|---|---|---|---|---|---|
| 4 | 6 | 36 | 36 | 36 | 128 | 4 |

Table 3.2: Proposed MT50 Bit Allocation For L5.

MT50 will deliver HMACs for the previous five messages and a hash point together. This proposed MT50 design requires eliminating the CRC. ER stands for erasure recovery. The ER fields enable the authentication even with the loss of two out of the five messages, and their computation is discussed in [108]. Moreover, [108] covers other designs that do not require eliminating the CRC and can work with L1 where $b_{\text{AMAC}} < 36$.

Section 3.1:

$$k_j = \text{HMAC}(p_i, t_j || \text{ PRN } || \text{Frequency}) \tag{3.10}$$

$$s_j = \text{TRUNC}\left(\text{HMAC}\left(k_j, m_j\right), b_{\text{AMAC}}\right) \tag{3.11}$$

$$k_{\text{AMAC}} = \text{HMAC}(p_i, \text{“MT50Key”} || t_i || \text{ PRN } || \text{Frequency}) \tag{3.12}$$

$$s_{\text{AMAC}} = \text{TRUNC}\left(\text{HMAC}\left(k_{\text{AMAC}}, s_{i-1} || s_{i-2} || s_{i-3} || s_{i-4} || s_{i-5}\right)\right) \; . \tag{3.13}$$

$t_j$ is the integer time of the authenticated message's broadcast and receipt. The pseudorandom noise code assignment (PRN) is the pseudorandom code assignment associated with the broadcasting geostationary satellite. Frequency is the frequency band of the particular transmission (e.g., a string containing L1 or L5). I discuss the necessity of the concatenation and HMAC operations of Eq. (3.10) in Section 3.1. Eq. (3.10) ensures that each HMAC for each message from each satellite has its own key. The output of each signing HMAC is truncated to the $b_{\text{AMAC}}$ most significant bits. The output of each KDF HMAC could be truncated to 128 bits but need not be. Table 3.2 included two additional $b_{\text{AMAC}}$-bit erasure recovery fields utilizing the method from [27] whose derivations are discussed in [108].

The authentication and erasure recovery from [108] works as follows. Suppose the receiver receives all five messages associated with an MT50. The receiver uses the hash point to compute the five corresponding $b_{\text{AMAC}}$-bit HMACs, and then uses the five to compute the AMAC. If the receiver-computed AMAC matches the commitment provided in the ***previous*** (see the following paragraphs and Fig. 3.19) MT50, then the five messages have message integrity within the TESLA framework. If the receiver does not receive a message among the five, it cannot compute all five $b_{\text{AMAC}}$-bit HMACs to compute the AMAC. However, provided the receiver lost only one or two messages among the five, it can recover the missing message $b_{\text{AMAC}}$-bit HMACs to compute the AMAC.

The net effect of this AMAC procedure is that (1) the security increases (compared to the five 16-bit HMAC MT50 design from [13]) and (2) all messages are lost if more than two messages are lost. Further study on other error correction codes (e.g., Reed Solomon) is ongoing at the time of writing this thesis.

At the receiver receipt of an MT50, the included AMAC corresponds to the five previous messages and a secret unreleased hash point known only to the SBAS provider at message sending time. The included 128-bit hash point corresponds to the AMAC included in the ***previous*** MT50 message sent six seconds earlier. Fig. 3.19 provides a conceptual diagram of the delayed hash point release cadence. The combination of $\Theta = 6$ and the five-group message authentication means the TTA will be between 7 and 11 seconds after their broadcast. To minimize integrity message authentication time, the SBAS provider should set the cadence of integrity messages to always precede the scheduled MT50s. If the receiver cannot authenticate a message because of a lost MT50, it generally disregards that message (information that alerts the receivers to decrease the level of trust need not be disregarded).

The hash point provided within a specific MT50 is not the hash point used to generate the HMACs within that same MT50. If that were true, then $\Theta = 0$, which is impractical. Instead, the HMACs of a particular MT50 are derived from the hash point delivered in the following MT50, inducing a $\Theta = 6$. Fig. 3.19 provides a diagram of this design effect.

### 3.4.3 SBAS Alerts

Per the SBAS specifications, in the event of a GNSS integrity alert, an alert message must be sent by the SBAS provider four messages in a row [79]. Therefore,

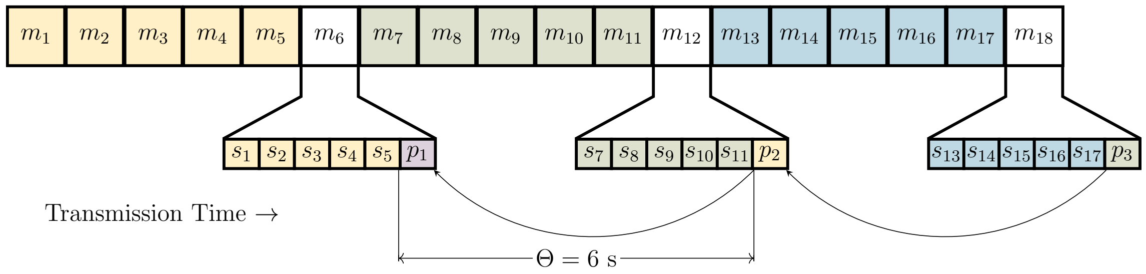


Figure 3.19: Conceptual Diagram Of How Consecutive Messages Relate In A Proposed SBAS TESLA Scheme.

The colors correspond to which messages and hash point produce which HMACs. The AMAC recovery scheme from Section 3.4.2 is not depicted. Instead the five 16-bit HMAC per MT50 scheme from [13] is depicted to simplify the diagram. Each MT50 includes the HMACs of the five previous messages, and the hash point used for those HMACs is sent six seconds later.

occasionally, an alert message will take priority over an MT50. To accommodate the perturbations of the schedule, [13] suggests a modified counter within HPF, as depicted in Eq. (3.14):

$$p_{i+1} = \mathrm{TRUNC}\left(\mathrm{H}\left(p_i || \,\mathrm{salt}\, || (t \oslash 6)\right), 128\right) \ . \tag{3.14}$$

In Eq. (3.14), $\oslash$ means integer division. Suppose that SBAS nominally transmits an MT50 message every six seconds when $\mod(t, 6) = 0$. If the MT50 is delayed 1-4 seconds, then $t \oslash 6$ remains unchanged without disrupting the hash path.

Nominally, the HMACs within an MT50 will derive from the following MT50. In an alert condition, the MT50 is delayed. Because delaying an MT50 delays the contained HMAC commitments, the contained HMAC commitments need to be signed with the following MT50. Otherwise, the $\Theta = 6$ condition breaks. This is depicted in Fig. 3.20.

Fig. 3.20 depicts seven scenarios in SBAS operation: nominal and six different alerts with different start times relative to the MT50 schedule. In Alerts 1 and 2, the alerts replace non-MT50 messages without causing a change in the transmission

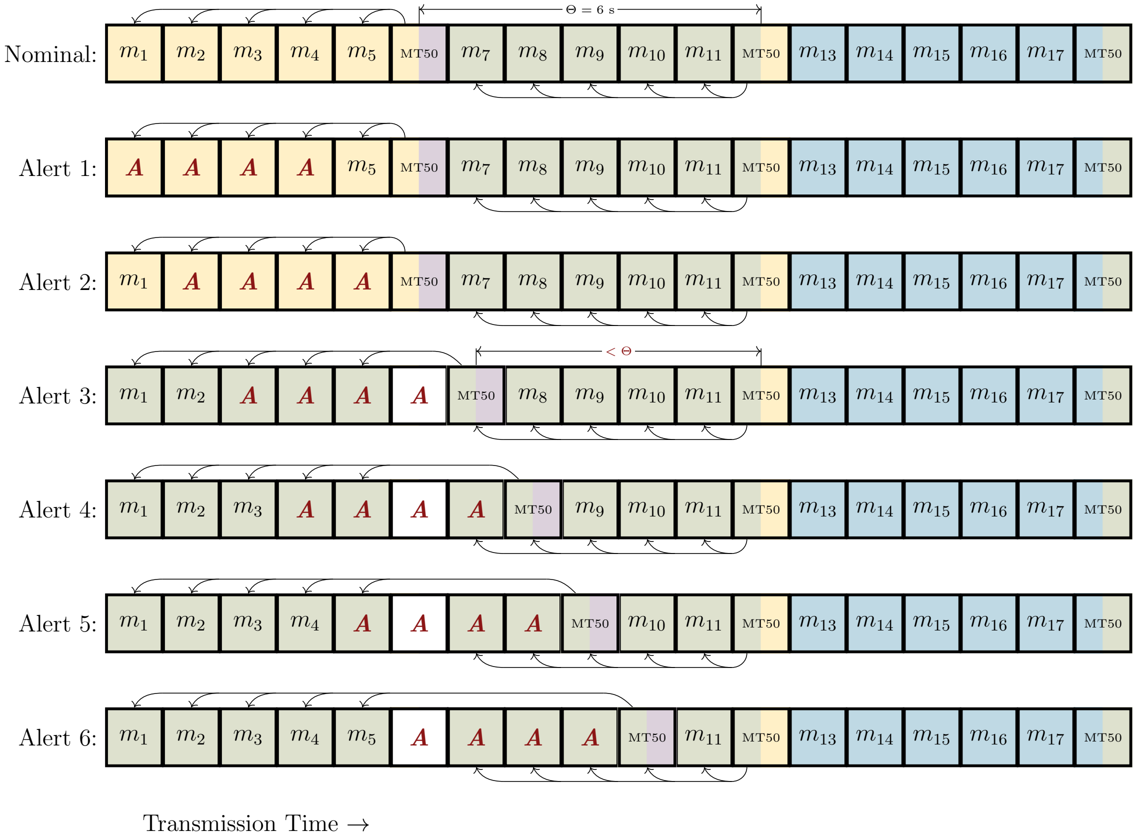

Θ = 6 s
Nominal:
Alert 1:
Alert 2:
Alert 3:
Alert 4:
Alert 5:
Alert 6:
< Θ
$m_1$ $m_2$ $m_3$ $m_4$ $m_5$ MT50 $m_7$ $m_8$ $m_9$ $m_{10}$ $m_{11}$ MT50 $m_{13}$ $m_{14}$ $m_{15}$ $m_{16}$ $m_{17}$ MT50
A
Transmission Time →

Figure 3.20: Conceptual Diagram Of How An SBAS Alert Affects TESLA Information Delivery.

The colors correspond to which messages and hash point produce which HMACs. The arrows correspond to which MT50 authenticated which messages. To satisfy the Θ requirement, in the Alert 3 through Alert 6 cases, messages 1 through 5 must be authenticated with the **green** hash point.

of the scheduled MT50. In Alerts 3 through 6, the alerts must supersede the scheduled MT50, causing the corresponding MT50 to be transmitted at a later time than scheduled.

In the Nominal, Alert 1, and Alert 2 cases, messages 1 through message 5 are signed with the yellow hash point. In the Alert 3 through Alert 6 cases, message 1 through message 5 must be signed with the **green** hash point. Because MT50 contains the commitments, its delay shrinks the boundary enforced by Θ, as depicted

| Preamble | MT | IOD | Page # | Transition | HMAC1 | HMAC2 | ... | HMAC6 | spare | CRC |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 6 | 4 | 4 | 6 | 32 | 32 | ... | 32 | 10 | 24 |

Table 3.3: Proposed MT53 Bit Allocation For L5.

MT53 will deliver HMACs for the navigation message (NM) of the core constellation GNSS constellations. The issue of data (IOD) information enables a receiver to associate the order of the HMACs to corrected satellites. The spare bits enable this message to work with both L5 and L1. With up to 92 satellites per mask, 16 pages are needed and 4 bits for the page number. Moreover, a transition indicator per satellite is needed to indicate whether the authenticated information is covering a transition of the information.

above the stream of Alert 3 in Fig. 3.20. To accommodate, simply having each of those messages be signed by the following hash point ensures that the $\Theta$ requirement is satisfied. Since the HMAC keys are derived with KDF via Eq. (3.10), the input hash point need only be incremented.

To simplify the authentication logic, in all cases, each MT50 authenticates its originally scheduled messages. For Alerts 3 through 6, one message is left unauthenticated per alert sequence (the uncolored message 6 occupying the original MT50 slot in Fig. 3.20), and the delayed MT50 is redundantly authenticated by the commitments in message 12. Since the alert information in that unauthenticated message does not need to be authenticated before its use (because alerts tell receivers not to use SBAS at all), the unauthenticated message 6s do not pose a security threat to the receiver.

### 3.4.4 MT53: Core Constellation NMA with SBAS

Another feature accessible with SBAS TESLA is utilizing SBAS to authenticate the navigation messages from the core GNSS constellations. Table 3.3 provides a proposed definition for a message that delivers core constellation NMA under SBAS TESLA: MT53. Each MT53 will deliver 6 NMA 32-bit HMACs to provide core constellation NMA.

Providing core constellation NMA with SBAS leverages a main feature of TESLA: its bandwidth efficiency. The HMACs can derive from the same hash path already distributed by MT50. Eqs. (3.15) and (3.16) provide the formulae to generate the HMAC keys and HMACs:

$$k_j^{\mathrm{NM\ PRN}} = \mathrm{HMAC}(p_i^P, t_j || \mathrm{PRN}_{\mathrm{SBAS}} || \mathrm{Frequency}_{\mathrm{SBAS}} || \mathrm{PRN}_{\mathrm{Core}}) \tag{3.15}$$

$$s_j^{\mathrm{NM\ PRN}} = \mathrm{HMAC}(k_j, \mathrm{NM}_{\mathrm{PRN}}) \,. \tag{3.16}$$

Care must be taken to ensure the distribution of the commitments in MT53 satisfies the $\Theta$ requirement.

The accompanying IOD[2] enables the receiver to associate the six HMACs to the correct satellites. Since MT1 and MT31 already provide an IOD mask of the most relevant satellites currently under correction of a particular SBAS, I propose that the order of the six HMACs correspond to the order prescribed in the mask with the corresponding IOD already provided to the receiver. Within Table 3.3, there are bits to indicate page numbers and NM transition (i.e., the case where the particular message is signing a previous message during a NM transition period).

The choice of six 32-bit HMAC enables $2^{-32} < 10^{-9}$ security without some of the MAC accumulation strategies (discussed in Section 3.4.2 and [38]. MT53 could be further improved with advanced error correction strategies, like those mentioned in Section 3.4.2.

[2] IODM for L5 corresponding to the mask in MT31 and IODP for L1 corresponding to the mask in MT1.

# Chapter 4

# GNSS TESLA Maintenance

> [P]rinters and paper ship differently. It would be faster to deliver them separately.
>
> Darryl Philbin, *The Office*

Chapter 3 discussed how to use geometry to construct useful authentication structures to minimize data bandwidth and/or time to authentication (TTA). All of these hash path geometries must end at a hash point signed with an asymmetric digital signature (DS) and that hash path end (HPE) is not used to sign the message stream. Instead, the HPE, the accompanying DS, and any DSs of DSs serve as infrequent maintenance for the Timed Efficient Stream Loss-tolerant Authentication (TESLA) scheme. This chapter is about how to design the distribution of that infrequent maintenance[1].

Because the HPE is accompanied by a DS, its authentication does not rely on the timing of its transmission (unlike the other hash points). Fig. 4.1 provides a conceptual diagram of how a Global Navigation Satellite System (GNSS) can distribute this maintenance information. The diagrams of Chapter 3 were in order of time of release by the constellation, just like the top part of Fig. 4.1. The DS and HPE were

[1] This chapter is based on my two publications regarding efficient TESLA maintenance schemes for SBAS [13, 14].

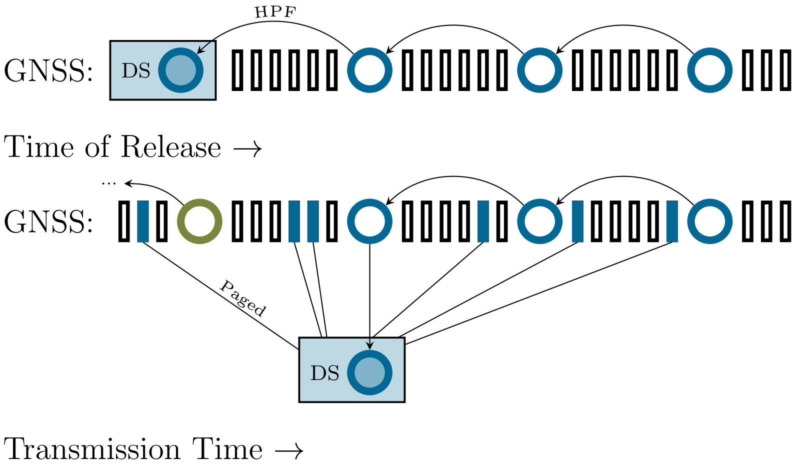


Figure 4.1: Conceptual Diagram Of TESLA Maintenance Information Distribution.

The top diagram is in order of time of release like the other diagrams of Chapter 3, reflecting how the HPE and the accompanying DS are released first. The bottom diagram is in order of transmission time to reflect several practical concerns of the GNSS constellation. The GNSS constellation will want to distribute the TESLA maintenance information before the hash path's use for continuous operation by always-on receivers. The diagram shows the last hash point of a transitioning-out hash path (**green**). For the transitioning-in hash path (**blue**), the diagram shows the first hash point, the HPE, and the maintenance information distributed in the message stream. The GNSS constellation will also want to distribute the TESLA maintenance information repeatedly in the stream to allow cold-start receivers that missed the initial transmission to operate. The DS on the information means the information need not be distributed on a strict schedule; rather, it can be distributed in the normal message stream where the data bandwidth allows. Because this maintenance information is much larger, it must be distributed into pages.

represented to the far left (earliest) part of the diagram because they are released first.

The bottom part of Fig. 4.1 is in transmission time to reflect two practical and necessary concepts. First, the constellation should transmit the DS and HPE before the hash path's use so that receivers can utilize the TESLA hash path immediately for a smooth transition between hash paths when preimage hash points run out. Second, the constellation should transmit them repeatedly to help cold start receivers initialize authentication when they did not receive it during its original transmission.

Because the HPE, DS information, and other scheme maintenance (e.g., salt) is large, most constellations will need to transmit this information in pages, as depicted in Fig. 4.1. The speed with which the constellation can deliver all of the pages determines the time to first authenticated fix (TFAF), an important performance indicator for the GNSS constellation. The explicit separation of the TESLA ***maintenance*** concepts of this chapter from Chapter 3 reflect that these concepts are better designed separately. Whereas the cadence of hash point distribution affects the TTA, the cadence of the maintenance information affects the TFAF. And decoupling their design will help with performance indicators overall.

## 4.1 No-Maintenance Designs

To design a scheme with absolutely zero maintenance burden, the GNSS must receive the maintenance information directly from an external network. This would be impractical for disconnected receivers unless large amounts of maintenance information are stored in an encrypted state until use. For instance, suppose the receiver user base is expected to undergo maintenance periodically, like with the 56-day cadence specified by the International Civil Aviation Organization (ICAO) Aeronautical Information Regulation and Control (AIRAC) schedule [78]. Then, during the maintenance, the receiver downloads the next several HPEs covering until the next expected receiver maintenance session.

Release of the HPEs early is secure due to the hiding commitment property from the ***salted*** hash path function (HPF). There are two safe ways to distribute the salt to maintain the most effective hiding property. First, the GNSS constellation can distribute the salt in the message stream authenticated by the previous hash path immediately before its use. Second, the receiver can download the salt in an encrypted state, and the constellation releases the salt-decryption key in the message stream authenticated by the previous hash path immediately before its use.

Both of these aforementioned options satisfy the necessary conditions on the salt to protect the hiding commitment property of HPF. The salt must be authenticated to prohibit tampering by an adversary. If an adversary can tamper with the salt's message integrity, the adversary could engage in a Rainbow Table Attack with the adversary's chosen salt to forge messages. Hence, it must be authenticated via DS if downloaded or via TESLA if received via GNSS. The salt must be unknown to everyone (except the authentic GNSS provider) until immediately before its use. Storage in the encrypted state preserves the confidentiality of the salt before release of the decryption key.

A no-maintenance design alleviates data bandwidth but requires additional data storage for the receiver and GNSS —though memory is cheap. GNSS will need to precompute the hash paths, which could be hundreds of thousands of hash points, and ascribe them times in the future (since HPF is a function of time). Essentially, GNSS would commit to the hash point messages far in advance. And, if a hash path is compromised, it would prohibit all receivers from accessing authentication until they receive maintenance to download the new HPEs.

Each stored HPE and encrypted salt must be authenticated by the GNSS's root certificate. Naively, GNSS could require that individual pieces of information have an individual root certificate DS. For intuition, the DS itself will likely be larger than the authenticated information itself. However, the GNSS can use a Merkle tree to cut down the storage requirement [65].

Incorporating the stored authenticated information into Merkle Trees allows the number of total DSs to decrease. The authenticated information is committed into a tree structure, where combinations of commitments receive a DS. Because of the commitment function's security, all information within the tree structure can be authenticated with a number of signatures that grows with the log of the number of objects authenticated.

While providing HPEs in advance as stored information would pose the greatest storage requirement among the suggestions of this chapter, Merkle Trees can be used for other object storage, such as certificates and backup certificates (see Section 4.3.2). Galileo has incorporated Merkle Trees into its prestored certificate design [39].

## 4.2 The Problems with Revocation

Incorporating certificate and hash path revocation features poses considerable burdens. For the case of a compromised certificate, there is no way to provably be assured that once GNSS issues a certificate revocation, all receivers will not be susceptible to attack.

Suppose that the GNSS has a single-bit indicator that indicates the specific certificate (or the authentication scheme generally) is revoked. For instance, Galileo's NMAS field [39]. GNSS could include this bit under a DS, but the certificate is compromised, so it makes sense for this bit to be unsigned. If this bit is unsigned, this lends to a substantial probability that noise could brick the receiver if that bit were ever flipped. Or, it lends to an easy way for a spoofer to brick the receiver by spoofing that bit.

When a cryptography secret is compromised, cryptography cannot help you. If one certificate is compromised, they all could be. GNSS could maintain isolated backup certificates (i.e., the secrets were stored separately and therefore avoided the initial leak) that are installed on receivers for this scenario. The receiver could know that if a hash path terminates at an HPE signed by a backup certificate, then the primary certificate has been revoked. However, the adversary can simply jam this indication and transmit forgeries with the compromised certificate.

Designing a provably safe revocation feature is a losing game of cat and mouse. However, incorporating revocation features can alleviate some of the risks with compromised certificates. Presumably, the spoofer with the compromised certificate would not be able to prohibit the revocation for most receivers. And perhaps one of the revocation notices will go through in the event of an attack. Or, if a GNSS designer elects a design like that from Section 4.1, then the network infrastructure needed for that design could accommodate emergency revocation scenarios to help receivers recover quickly.

## 4.3 Maintenance Design Management

Recall from Section 1.3.1 that asymmetric cryptography involves public keys, private keys, and DSs. The private key generates a DS on the authenticated information, and receivers use the public key to verify a DS. These cryptographic objects are managed under public key infrastructure (PKI). PKI includes additional practical concerns, such as cryptoperiod expiration and identity. The public key and associated metadata collection is standardized and called a digital certificate. There can be multiple levels of digital certificates (i.e., a stronger digital certificate authenticates a weaker and transient digital certificate). Usually, the top-level certificate is issued by a certificate authority (CA). A CA is usually an international or intercompany organization responsible for managing PKI.

In the diagram of Fig. 4.1 and others like it, only the HPE and the DS are represented for brevity. If the GNSS elects not to utilize a scheme from Section 4.1, then the maintenance information must include each of the following:

(1) hash path end (HPE);

(2) hash path salt;

(3) information that enables a receiver to relate each hash point to the specific release time (e.g., the time the first hash point is released and release cadence schedule) unless the scheme rigidly specifies that (e.g., hash paths transition on a specific day and time per the interface control document (ICD)); and,

(4) a DS covering each of the above from the authentic GNSS provider.

The maintenance information can optionally include other information, depending on the design of the maintenance scheme. That information depends on whether the scheme relies on pre-installed information (see Section 4.3.2) or must rely on over-the-air rekeying (OTAR) (see Section 4.3.1).

The DS must cover the HPE to complete the TESLA authentication security argument from Section 1.3.4. Without incorporating the HPE, the adversary could simply generate a consistent hash path to cause a receiver to accept forged messages.

| Protocol | Public Key Bits | Signature Bits |
|---|---|---|
| ECDSA | 257 | 512 |
| EC Schnorr | 256 | 384 |
| Dilithium | 10560 | 19360 |
| FALCON | 7176 | 5328 |
| Rainbow | 1262400 | 528 |

Table 4.1: Bit Counts For Maintenance: 128-Bit Security [69].

| Protocol | Public Key Bits | Signature Bits |
|---|---|---|
| ECDSA | 513 | 1024 |
| EC Schnorr | 512 | 768 |

Table 4.2: Bit Counts For Maintenance: 256-Bit Security.

Next, the DS must cover the hash path salt. Otherwise, the hiding commitment property is not assured (see Section 1.3.1), and the adversary could engage in a Rainbow Table Attack to discover a consistent hash path to cause a receiver to accept forged messages. Lastly, unless the scheme rigidly specifies the schedule of hash path rotations, the DS must cover the applicable time and cadence of the hash path. Otherwise, the adversary could utilize an authentic GNSS-generated hash path with the HPE signed by a DS and use them later to cause a receiver to accept forged messages.

In addition to the aforementioned objects, the GNSS may elect to include additional maintenance information depending on the desired features (e.g., key revocation) and the desired PKI structure. For instance, the PKI structure could include several layers of public key digital certificates (e.g., with stronger security levels in effect for longer periods of time). Tables 4.1 and 4.2 provide the data bandwidth burden for various protocols, including some quantum-resistant protocols [69]. Depending on the GNSS system constraints, the GNSS may elect to require receivers to have preinstalled information or design an OTAR protocol.

The numbers within Tables 4.1 and 4.2 are approximate with respect to what is available among the standardized protocols. For instance, among the 256-bit-security Elliptic Curve Digital Signature Algorithm (ECDSA) curves, the National Institute of Standards and Technology (NIST) standardized a 521-bit curve. The choice of 521 over 512 relates to advantages with implementation speed and the values of known prime numbers. Moreover, standards include additional bits related to the expiration and encoding of a particular instance of the protocol.

### 4.3.1 Smooth Transition with OTAR

The TESLA maintenance schedule information will primarily have two goals: (1) an acceptable TFAF, and (2) a smooth transition (i.e., continuous authentication) when transitioning hash paths. In support of smooth transitions between hash paths, the GNSS constellation should deliver the maintenance information of the next hash path concurrently with the maintenance information of the current hash path. Albeit, the maintenance of the next hash path can be distributed at a smaller frequency. And to protect the hiding property of the hash path, the salt transmission should be distributed as close to before the use of the hash path as acceptable. Fig. 4.2 provides a conceptual diagram of this transition.

Fig. 4.2 provides a conceptual diagram depicting the transition of three hash paths, depicted **green**, **blue**, and **orange**. The hash path itself is depicted in a darker hue, whereas the associated maintenance is depicted in a lighter hue within the message stream. The colors within the message stream represent relative frequency rather than imposing an exact schedule. There are four periods marked. During Period A, the **green** hash path is active. **green** and **blue** maintenance information is broadcast, but the **blue** maintenance information is broadcast at a lower frequency. The higher **green** frequency enables the constellation to meet its TFAF requirement for the **green** hash path. The lower frequency **blue** enables receivers that are usually on to eventually accumulate the necessary **blue** maintenance to support a smooth future transition to the **blue** hash path, even if they are turned off during Period B.

Period B simultaneously broadcasts the **blue** and **green** maintenance information to meet the **blue** and **green** TFAF requirements. After the last preimage for the **green** hash path is released, the constellation must start using the **blue** hash path for authentication. In Period C in Fig. 4.2, the constellation only broadcasts the **blue** hash path maintenance information because the upcoming **orange** hash path

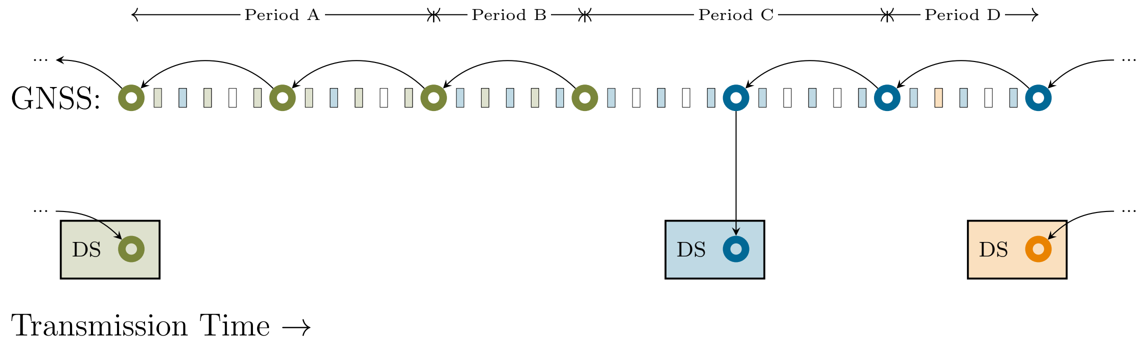


Figure 4.2: Conceptual Diagram Of Hash Path Transition And The TESLA Maintenance In The Data Stream.

Depicted are hash points (dark shade), paged TESLA maintenance information (light shade) over the use of three hash paths: **green**, **blue**, and **orange**. The diagram is abbreviated: a more accurate diagram would involve hash paths on the order of 1000s of hash points; and, the periods marked would involve 10s-1000s of hash points. Moreover, the light-shade messages indicate only relative frequency of the indicated hash path maintenance. A realistic diagram would not allocate 80%-100% of the data bandwidth to TESLA maintenance; rather, a much smaller portion of the data bandwidth is allocated to maintenance and the other bandwidth is allocated to other information (e.g., ephemerides, corrections). Section 4.3.1 describes the meaning of each of the parked periods.

is so far in advance. At some point within Period D, the constellation returns to the frequency of Period A, except with **blue** and **orange** maintenance information.

Fig. 4.2 depicts the intuitive design consideration and choices a GNSS designer must make to accommodate their TFAF requirements. It does not prescribe a frequency. The GNSS could replace Period C with Peroid D. For instance, the constellation could always broadcast the current hash path maintenance at a fixed frequency and the next hash path maintenance at another fixed frequency. In [13], the maintenance data for a single hash path requires about 17 paged messages. The current hash path maintenance is broadcast approximately one out of every 17 messages, and the next hash path maintenance is broadcast one out of every 289 messages. That cadence supports a TFAF of less than five minutes, and if the receiver is on continuously for one hour, it will have the needed information for the next hash path transition.

### 4.3.2 Pre-installed Maintenance

The receiver must have some preinstalled information. At a minimum, a root certificate issued by a CA (or GNSS itself) establishes the Root of Trust (RoT). An authentication scheme cannot exclusively rely on certificates distributed in the TESLA maintenance message stream because it must associate the identities within the certificates to GNSS. Either these identities must be authenticated using a CA, which is standard practice in the internet community, or GNSS must affix those certificates in ICD-like documentation.

The simplest possible design would be for GNSS to engage in no OTAR. All certificates are preinstalled on the receiver, and the message stream only delivers DSs based on those fixed certificates. The main issues with designing a system like this are the cryptoperiod and data bandwidth. Typically, GNSS are designed with a large, unchanging service guarantee. Current standards recommend a 1-2 year cryptoperiod for public key authentication methods [102]. Therefore, long-term certificates likely need to use the 256-bit-security-level from Table 4.2, doubling the DS maintenance data transmitted over the 128-bit-security-level certificates. If that is unacceptable, see Section 4.3.3.

### 4.3.3 OTAR With Intermediate Certificates

In keeping with the general recommendation that certificates have short cryptoperiods and be recurringly rotated, and to decrease the bandwidth required per hash path renewal, the GNSS designer can incorporate multiple levels of certificates. Fig. 4.3 compares two ECDSA schemes, though the general trend is the same regardless of the asymmetric authentication protocol. The left one utilizes a single, long-term certificate preinstalled on receivers to anchor TESLA hash paths. The right one utilizes two levels of certificates with a smaller intermediate certificate.

Suppose, for the sake of concreteness, that the GNSS designer elects to use a two-level scheme with the intermediate certificates rotating every month and a new hash path every day. And suppose the GNSS is willing to accept that the TFAF for a receiver turned off for longer than an entire month is 10 times longer than a receiver

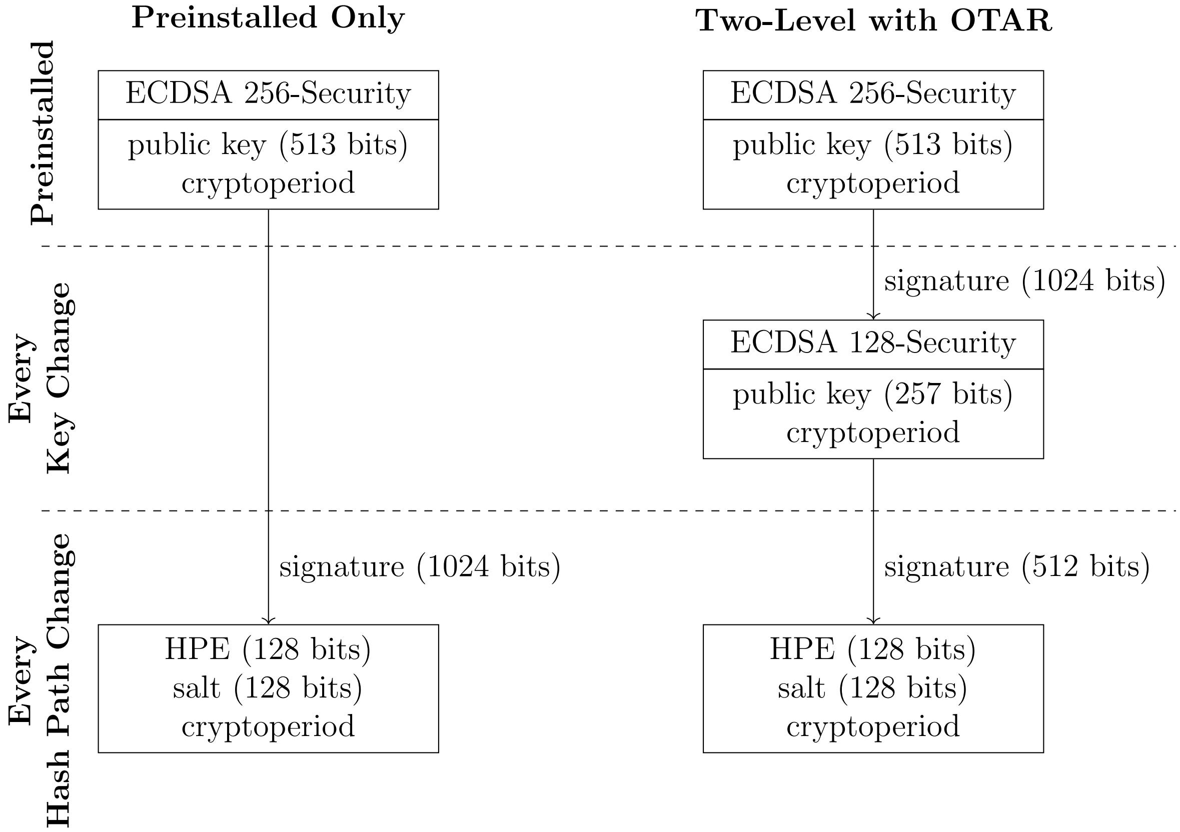


Figure 4.3: Conceptual Diagram Comparing The Data Bandwidth Requirements Of Two TESLA Maintenance Schemes.

On the left is a scheme with only a single, long-term certificate preinstalled on receivers. A total of approximately 1280 bits must be transmitted for each hash path change. On the right is a scheme with two levels of certificates with a smaller intermediate certificate with 128-bit security. A total of approximately 768 bits must be transmitted for each hash path change; however, 1281 bits must be transmitted for each intermediate certificate change. By having those 1281 bits transmitted less frequently, the two-level scheme could enable receivers that turn on and off frequently to have a faster TFAF at the expense of the TFAF for receivers that are turned off for prolonged periods.

[tbp]

| Parameter | Description |
|---|---|
| $G$ | Elliptic curve base point |
| $n$ | Integer order of $G$ |
| $k$ | Cryptographically secure random integer that is different for every signature |
| $C$ | A point computed on the elliptic curve |

Table 4.3: ECDSA Notation From Wikipedia [110].

that has been on in the last month. Let $T$ be the TFAF for the left scheme from Fig. 4.3, and let $\alpha$ be the advantage for the TFAF for receivers that are not off for more than a month. With a fair comparison, the data bandwidth for the TESLA maintenance is the same; therefore, using the numbers from Fig. 4.3,

$$\frac{1280}{T} = \frac{768}{\alpha T} + \frac{1281}{10 \alpha T} \tag{4.1}$$

$$\alpha = 0.7 \ . \tag{4.2}$$

Hence, receivers that are on frequently enough to know the intermediate certificate can have approximately a 30% reduction in TFAF at the expense of receivers turned off for a month.

### 4.3.4 ECDSA-derived Hash Path Salt

It is possible to securely deliver salt without delivering it with the other pages of information with some DS protocols. For an ECDSA DS, this can be done by having the salt derived from a cryptographic object within the ECDSA protocol. Essentially, the salt is derived from the DS itself, thereby alleviating the data bandwidth required for the salt. For this section only, I will use the notation available on the ECDSA Wikipedia page [110], which is reproduced in Table 4.3.

Within the ECDSA protocol, a cryptographic nonce is drawn to generate the signature. As an aside, this also makes multiple ECDSA signatures on the same message different. If any of the cryptographic nonces were leaked or predictable, including if the nonce were used multiple times, then the ECDSA private key is

**Algorithm 4.1:** Transmitting Salt Without Additional Messages With ECDSA.

```
Provider
Provider generates cryptographically-secure nonce $k$
With elliptic curve base point $G$, Provider computes elliptic curve point
  $C = k \times G$
salt $= H(C)$
Provider computes hash path $p_0 ... p_L$ with Eq. (1.2)
Provider generates ECDSA signature for $p_L$ with $C$ and broadcasts before
  actual use of hash path
Receiver
Receiver receives $p_L$ with ECDSA signature for authentication
Receiver derives $C$ from ECDSA signature
salt $= H(C)$
Receiver authenticates new message on a new hash path upon receipt of $p_n$.
```

compromised. To distribute the salt via the DS, the GNSS must derive the salt from this nonce. This means that the GNSS must start the ECDSA protocol by drawing a nonce and deriving a salt from it, then compute the entire hash path, and then finally generate a DS on the HPE. The procedure is delineated in Algorithm 4.1.

When the signature is released, the salt is released. Therefore, the GNSS provider should be aware that the earlier the salt is released, the less secure the hiding commitment property on HPF. Releasing a week before its use is probably fine, but perhaps not a year before its use [32].

While this scheme poses the advantage of less maintenance data bandwidth, it hinders the scheme's flexibility. Using a nonce more than once reveals the secret private key used to authenticate the data. This means that only one DS can be distributed on the hash path, eliminating the possibility of the feature in Section 4.3.5

### 4.3.5 Resigning Intermediate Hash Points

Suppose that the hash path length $n$ is very long, perhaps on the order of 100000 hash points. Compare the two cold-start receivers from Fig. 4.4: Receiver A turns on immediately before the transition of a hash path and Receiver B turns on immediately

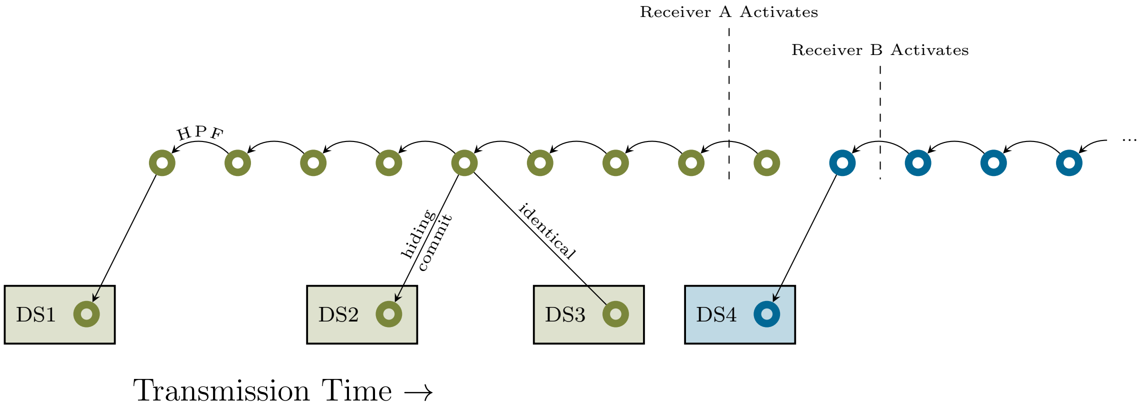


Figure 4.4: Conceptual Diagram Depicting The Advantage For Recurring DS On Hash Points Within Hash Paths.

> There are two hash paths over a transition: **green** and **blue**. Suppose the GNSS constellation distributes only one DS per hash path: in the diagram, DS1 and DS4. When cold-start Receiver A activates, it must hash all the way to the HPE signed by DS1; whereas, cold-start Receiver B must hash substantially fewer times to get to the HPE signed by DS4. When hash paths are very long, the start-up computation effort for Receiver A may be a material concern for the GNSS designer. The GNSS may elect to distribute digital signatures of hash points mid-path to save computational efforts for Receiver A. Whereas DS1 and DS4 may be distributed before use because the covered hash points are ***not*** used for CMTs, DS3 (and other DS that sign hash points mid-path) ***must*** be transmitted after use to generate CMTs. To bypass this, one could make a hiding commitment (that is not with HPF), as for DS2.

after the transition of a hash path. Suppose the GNSS TESLA scheme only contains DS1 and DS4 from Fig. 4.4. DS1 and DS4 follow what has been shown before: the HPE is covered under a DS, which is transmitted before the hash path's use. Upon startup, Receiver A will require a much larger start-up computational effort to authenticate the first message than Receiver B. While Fig. 4.4 does not depict very many hash points, this can be a computational concern for receivers with longer hash paths [30].

To assist receivers with unfavorable start-up times with respect to the hash path transition, the simplest solution is for GNSS to provide DSs of hash points in the middle of the hash path. This is depicted with DS3, where the ***expired*** hash point is redistributed with a DS in the maintenance data bandwidth. This DS ***must*** be distributed after its use to authenticate under TESLA; otherwise, a preimage hash point would be released early, violating security.

If the GNSS wants to provide a hash-path-end-type commitment mid-path before its use, then it should be made with a hiding commitment. HPF provides a hiding commitment with the incorporation of salt; however, it is already being used for the hash points. Because HPF does not include a changeable parameter (similar to key-derivation function (KDF)), the hiding commitment must be done with another function. A salted KDF would work provided the KDF function is also a commitment function, such as keyed-hash message authentication code (HMAC). This is depicted in Fig. 4.4 with DS2.

### 4.3.6 Page Transmission Optimization

Similar to concepts discussed in Section 3.2, GNSS can act as a parallel channel for maintenance information distribution to decrease TFAF. When the maintenance information is the same for the entire constellation, the distribution time can be decreased by a factor on order with the number of satellites and frequencies. The system could divide the maintenance information into pages and then transmit random pages on the channel to leverage this improvement. For a single channel, distribution performance decreases according to the Coupon Collector's Problem.

For compatibility with receivers that only track a single channel (i.e., single satellite and frequency), the pages should cycle through a fixed sequence. To increase the distribution performance, individual channels can distribute the pages out-of-phase. For instance, from [13], the six Wide Area Augmentation System (WAAS) channels (three satellites and two frequencies) can distribute the TESLA maintenance out of phase. This idea is being adopted by Galileo for its Open Service Navigation Message Authentication (OSNMA) information distribution without interfering with the existing OSNMA ICD [33, 34, 41].

Error correction codes could also be helpful, such as Fountain Codes [43]. A Fountain Code requires additional pages to distribute the TESLA maintenance information. However, receipt of a specific number of pages is required for transmission rather than a particular set of pages. Fountain Codes provide an error correction code that behaves like a water fountain filling a cup: the receiver can intermittently fill its cup until it's full, the same way it would assemble the pages with a fountain code.

## 4.4 Application to SBAS

In Section 3.4, I review the strategies of Chapter 3 applied to a Satellite-based Augmentation System (SBAS) in [13]. In this section, I review the strategies of this chapter applied to SBAS in [13].

In [70], they suggested appending a single message type (MT) to the SBAS schedule, MT50 for L5 and MT20 for L1. Within [70], the MT50 contains 26 bits for the scheme's maintenance. A large number of MT50s would be needed to deliver all of the maintenance information. To improve the TFAF and make the delivery more robust to loss, [13] suggests that a separate message handle TESLA maintenance information: MT51. By separating the TESLA authentication information delivery from the TESLA maintenance information delivery, the two aspects of the scheme can separately better meet performance considerations.

Like this work, [13] is an academic exploration. The MT51 of [13] reflects its academic purpose by exploring what features are possible and what designs are elegant. In the process of standardizing MT51, many of the features have been eliminated. For this work, I review a stripped-down MT51, noting that final standardization will have some additional bit-level differences reflecting the preferences of the authors of the standards.

### 4.4.1 MT51: TESLA Maintenance Delivery

By separating the TESLA hash point and HMAC delivery with MT50 (see Section 3.4.2), the TFAF can be substantially reduced. Table 4.4 provides a proposed definition of MT51 with the bit allocations. The SBAS will send MT51 containing the current hash path's maintenance information frequently enough so that a receiver's

| Preamble | MT | Page Number | Payload | Spare | CRC |
|---|---|---|---|---|---|
| 4 | 6 | 6 | 200 | 10 | 24 |

Table 4.4: Proposed MT51 Bit Allocation For L5.

MT51 will deliver all TESLA maintenance information for current and future hash paths. The spare bits enable this message to work with both L5 and L1. This design is simpler than the one found in [13], reflecting the preferences of recent MT51 standardization efforts.

TFAF is less than five minutes. Moreover, the SBAS will send the next hash path's maintenance information frequently enough so that a receiver continuously operating for one hour will receive it. Any remaining empty slots within the SBAS schedule will be filled with MT51, delivering the current hash path maintenance for a faster TFAF.

The MT51 of this section is simpler and contains fewer features than the MT51 design from [13]. This reflects the movement of current MT51 standardization efforts. Section 4.4.1 contains a large payload section and a page number to allow receivers to associate pages. Once a receiver has received each of the pages, it will have the DS and other maintenance information to begin the TESLA authentication procedure with MT50. With the information from Section 4.4.3, the receiver must receive 11 unique pages.

### 4.4.2 ECDSA Certificate Structure

I propose a two-level ECDSA certificate structure, as suggested in [70]. I call them Level 1 and Level 2. Level 1 signatures authenticate Level 2 certificates, and Level 2 signatures authenticate TESLA HPEs and other hash path maintenance.

Level 1 keys will be 256-bit-security ECDSA keys managed internationally by a trusted CA. Each Level 1 key will be in use for 100 weeks. The CA shall compute a large number of Level 1 keys for use in the perpetual future and then encrypt each key individually via AES-128 with different AES encryption keys (one per Level 1 key) held secret by CA. The CA will distribute the AES-encrypted ciphertext to receiver manufacturers for pre-installation. As Level 1 keys expire, the CA will distribute the keys to decrypt the AES-ciphertext, one at a time, for the SBAS provider to distribute via MT51. As the receiver receives the key to decrypt its onboard AES-ciphertext, it will update its current Level 1 ECDSA public key. Each Level 1 public key will be 512-bits, and any signature derived therefrom will be 1024-bits.

Level 2 keys will be 128-bit-security ECDSA keys managed by the SBAS provider. Each Level 2 key will be in use for 10 weeks. For a new Level 2 key, the SBAS provider will generate a secure-random private ECDSA key and its associated public key. The SBAS provider will submit the new public key to CA for a signature from CA's current level 1 key. The SBAS provider will then distribute that public key and the associated authenticating signature over SBAS. The receiver receives a new Level 2 public key and the authenticating signature, verifying the received new Level 2 public key with the associated decrypted level 1 public key.

The SBAS provider will use its Level 2 keys to authenticate TESLA HPEs, and keys derived from the TESLA hash paths will be used to authenticate the bulk of SBAS messages with HMACs. For all levels, the authenticating pseudorandom data delivered will accompany data (e.g., SBAS message preamble, MT, and other data) that must be sent per the definitions below. A particular signature must derive from the entire SBAS message(s) used to deliver that particular key. Concretely, when a level 1 key authenticates a Level 2 key, the level 1 signature must derive from the entire set of messages used to deliver the Level 2 key and the accompanying key expiration time. In other words, the level 1 signature must derive from the complete messages containing overhead data, not just the Level 2 key itself. If this does not happen, then the accompanying data, especially the key expiration time, will not be secured by the cryptographic primitives.

### 4.4.3 Maintenance Information and Delivery Schedule

With the certificate structure from Section 4.4.2, Table 4.5 provides a list and approximate bit count for all of the information. The entire set of maintenance information for a particular hash path is called a TESLA Maintenance Stack. The numbers listed are approximate because certificates can include some other, potentially optional,

| Information | Bits | Delivery |
|---|---|---|
| Level 1 Certificate | 512 | Preinstalled |
| Level 1 Decryption Key | 128 | |
| Level 1 DS on Level 2 Certificate | 1024 | |
| Level 2 Certificate | 256 | |
| Level 2 Certificate Expiration Time | | |
| Level 2 DS on All Items Below | 512 | |
| TESLA HPE | 128 | |
| TESLA Salt | 128 | |
| Hash Path Applicable Time | | |
| **Total Per TESLA Maintenance Stack** | **2176** | 5 min |
| Next TESLA Maintenance Stack | 2176 | 1 hour |
| Other SBAS TESLA Maintenance Stack | 2176 | 1 hour |

Table 4.5: TESLA Maintenance Information For SBAS.

A full TESLA Maintenance Stack of 2176 bits requires 11 unique MT51s. The amounts listed are approximate depending on which standard ECDSA curve and certificate convention is used. Moreover, the timing bits are not listed. They could be 32-bits to follow the UNIX timing definition, or they could be a similar amount depending on the timing or certificate convention. There are still spare bits available in Section 4.4.1, and increasing the message count to 12 will not materially affect the results of this section.

metadata, and because standardized ECDSA curves sometimes do not conform to the 128, 256, 512, and other base-2 numbers. For instance, a popular 256-bit secure ECDSA curve produces 521-bit signatures rather than 512-bit signatures. In the global context, SBAS providers may want to employ ECDSA curves developed by their own country.

For a TESLA Maintenance Stack of 2176 bits, 11 MT51s are needed. To meet a TFAF of five minutes, assuming no message loss, an MT51 of the current TESLA Maintenance Stack must be sent every 27 messages. To distribute the next TESLA Maintenance Stack in an hour, an MT51 must be distributed 1 every 327 seconds.

In support of coordination between SBAS systems, an SBAS can deliver neighboring SBAS TESLA Maintenance Stacks at the same low frequency as its next TESLA

Maintenance Stack. For example, consider WAAS and European Geostationary Navigation Overlay Service (EGNOS). For an aircraft to transit between the WAAS and EGNOS service volumes, it must travel several hours in open sky conditions over the Atlantic Ocean. Even with a low delivery frequency of an hour, that should be fast enough to deliver the entire TESLA Maintenance Stack before the aircraft arrives at the other service volume, meaning this aircraft will have a TFAF of six seconds after it starts tracking the other SBAS (compared to five minutes otherwise).

If the SBAS provider separates each of the objects of Section 4.4.1 with additional padding and pages, then the objects can be transmitted at separate frequencies. In [13], I suggest that the cryptoperiod of the level 1, level 2, and TESLA hash paths be 100, 10, and 1 week, respectively. Feasible scheme designs could allow level 1 cryptoperiods to be years longer, level 2 cryptoperiods to be as short as a month or week, and hash paths to rotate every hour or day. Depending on the state of information stored on the receiver delivered by MT51, the receivers' TFAF upon startup will vary. The transmission frequency of individual objects can be optimized to assist receivers with specific operational cadences, as suggested in Section 4.3.3.

When every SBAS satellite and signal frequency share the same hash path and TESLA maintenance information, TFAF can be improved further for receivers that track multiple satellites and frequencies. The SBAS can transmit the individual MT51 pages out of phase among the satellites and signal frequencies. With three geostationary satellites and two frequencies, the TFAF savings would be on the order of 80%.

### 4.4.4 Validation with MAAST

In [13], I provide simulation data of the scheme therein that demonstrates that the authentication proposal does not adversely affect the service provided presently without authentication. The simulated data is generated by the Matlab Algorithm Availability Simulation Tool (MAAST). Fig. 4.5, adapted from [13], shows the MAAST-simulated availability for WAAS with and without authentication. There are small differences in the availability of the service volume boundaries at high latitudes. However, due

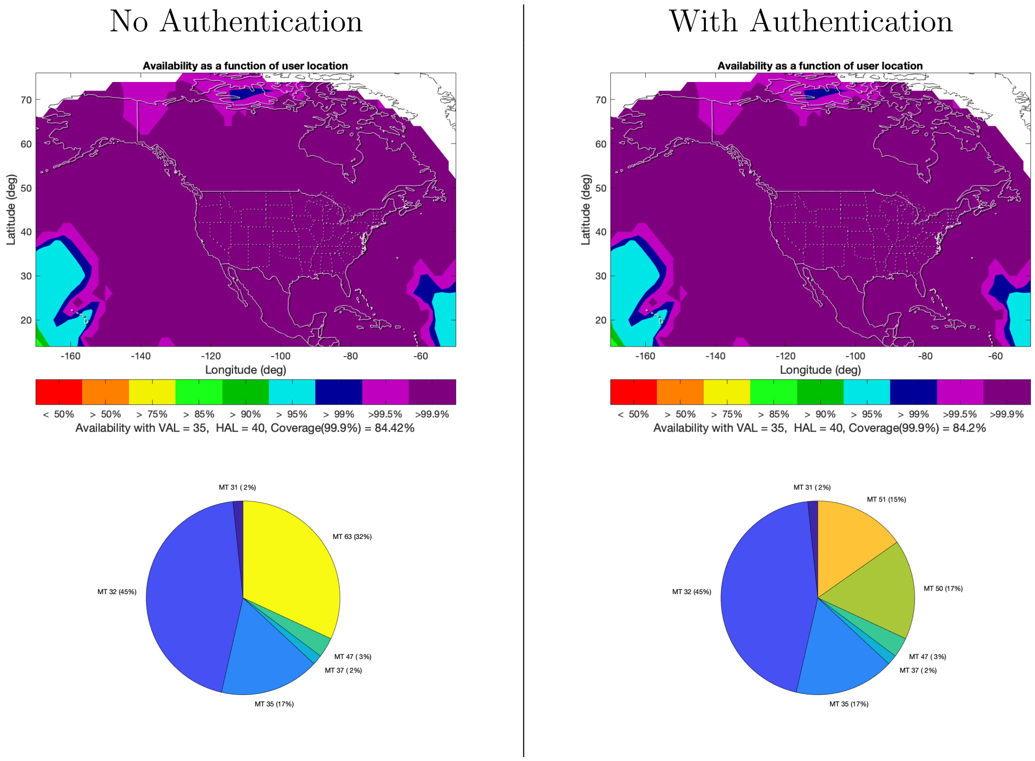


Figure 4.5: SBAS Authentication Availability Simulation Results.

In the figure, there are availability contour maps from the output of MAAST with no authentication (left) and with authentication (right). In the authentication case, messages were only accepted by the receiver after authentication. The pie graphs indicate the distribution of the different messages with no authentication (left) and with authentication (right). In the authentication case, MT51s are sent 1 out of every 6 messages. The remaining empty bandwidth is devoted to MT51 to further decrease the TFAF. There are some small coverage differences between the figures, such as north of Canada around $(-130°, 72°)$.

to the authentication data bandwidth using the unused data bandwidth, there is no expected degradation in the services currently available.

# Chapter 5

# Combinatorial Watermarking

> It is more important to have beauty in one's equation than to have them fit experiment.
>
> Paul Dirac

Recall from Section 3.1.2 that to enact ranging authentication, the Global Navigation Satellite System (GNSS) provider must have the ranging code include a hiding bit commitment. There are two ways of doing this. First, the entire ranging code can derive from a pseudorandom function (PRF) within the Timed Efficient Stream Loss-tolerant Authentication (TESLA) framework. Second, the ranging code can include a hidden watermark [91]. Fig. 3.5 and Section 3.1.2 discuss how to use key-derivation function (KDF) TESLA geometry to generate PRF ranging codes and KDF-cryptographic material for watermarked ranging codes. This chapter discusses (1) how to take the KDF-derived cryptographic material to create watermarks and (2) the signal processing required to assert authenticity with a watermark[1].

Watermarking signal authentication solves a specific problem associated with PRF ranging codes. To perform the ranging measurement, depicted in Fig. 1.1, the receiver must know the ranging code: the receiver must correlate its recorded radio signal with

[1] This chapter is based on my five publications regarding combinatorial watermarking for GNSS [4, 5, 6, 9, 10].

a replica of the signal. With a PRF-ranging code, receivers initially do not know the ranging code because it is entirely a hidden bit commitment (usually below the noise floor). After the reveal phase, when the receiver receives the hash point from which the PRF-ranging code is derived, the receiver can go back into the past and perform the ranging measurement. Without prior knowledge of the ranging code, the receiver must wait to track the signal.

With a PRF-ranging code, the entire code itself is a hidden bit commitment. With a watermarked signal, most of the ranging code is known to everyone, and only a small portion is involved in a hidden bit commitment. This enables all receivers to track the signal. Receivers can ignore the existence of the authenticating watermark or perform additional signal processing after the hash point distribution to determine the signal's authenticity.

While the act of ranging with a PRF-ranging code provides authenticity, a watermark requires additional signal processing later. Watermarking the signal degrades the ranging code, making it slightly harder to track. In Fig. 1.1, this results in a decrease in the matching convolution peak during the initial signal-tracking phase. After the hash point distribution, the receiver can use the hash point to derive the watermark and then redo the convolution. Convolution with knowledge of the watermark should increase the convolution for an authentic signal. This is depicted in Fig. 5.1.

Fig. 5.1 depicts how the receiver must process the watermarked signal a second time to assert authenticity. While the receiver knows enough about the watermarked ranging code to track the signal at its broadcast, the watermark is hidden throughout the ranging code. For this chapter's watermarks, the watermark is a selection of inverted ranging chips from the public ranging code. In Fig. 5.1, each colored line within the ranging code represents an inverted chip.

The bottom two rows of Fig. 5.1 occur within the receiver's signal processing, with the bottommost row shifted right, indicating that that processing comes later. In the Signal Processing Tracking row, after each ranging code, the receiver will convolve its recorded signal with the public ranging code replica. This corresponds to how the convolution peak windows in that row vertically align right after a ranging code

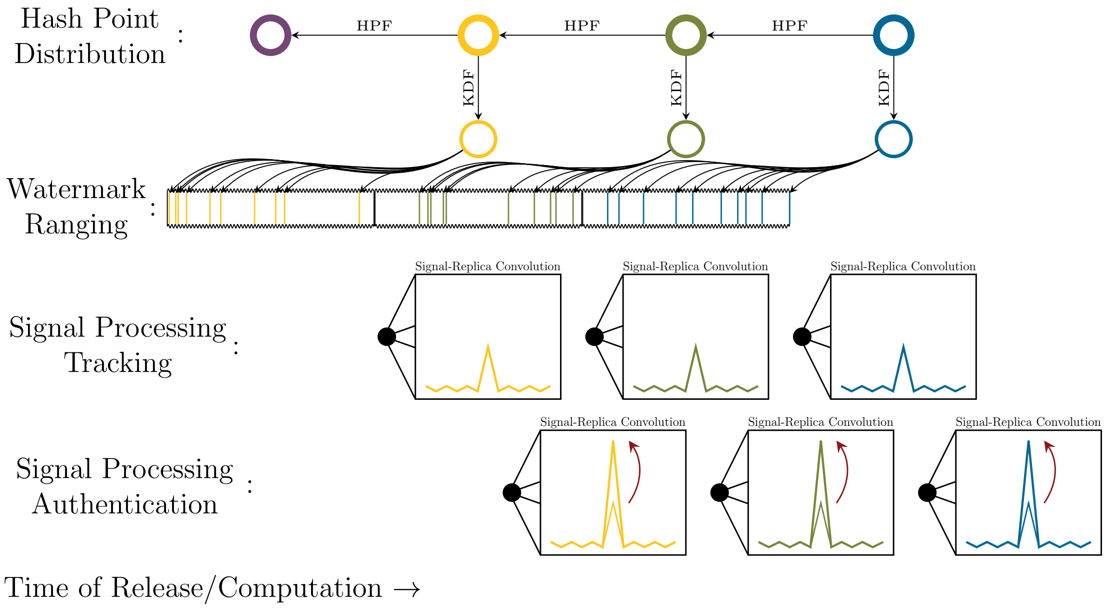

Hash Point Distribution :
HPF
HPF
HPF
KDF
KDF
KDF
Watermark Ranging :
Signal Processing Tracking :
Signal-Replica Convolution
Signal-Replica Convolution
Signal-Replica Convolution
Signal Processing Authentication :
Signal-Replica Convolution
Signal-Replica Convolution
Signal-Replica Convolution
Time of Release/Computation →

Figure 5.1: Conceptual Diagram Watermark Signal Authentication.

After each ranging code, the receiver computes the signal and replica convolution to find the peak to measure the satellite range. This is the same process that goes on everyday with today's non-authenticated signals. Because the signal is degraded by the watermark, the convolution peak is smaller. Later, after distribution of the hash point, the receiver goes back and reconvolves. This time, with the watermark incorporated into the receiver's replica. For authentic signals, the receiver should see an increase in the convolution peak. Adversaries should not be able to produce a signal with this peak-increase behavior because the watermark acts as a hiding bit commitment.

sequence in the row above. The receiver can track the signal from the lower peaks to produce Positioning, Navigation, and Timing (PNT) solutions. The convolution peak is lower than expected because the watermark inverts several of the chips. After the hash point distribution, when GNSS reveals the location of the inverted chips, the receiver can derive the watermarked replica and redo the convolution. This corresponds to how the convolution peak windows in the Signal Processing Authentication Row align to right after the hash point distribution in the top row. Observing an increase in the convolution peak indicates that the watermark signal is authentic.

A commonly suggested statistic derives from a matched filter with the correct security code sequence, for instance, as in [31]. In [31], the authors propose a hypothesis test on the post-correlation data that presumes the output is normally distributed in both hypotheses and use experimental data to determine those normal distributions. Then, they use the Kullback-Leiber divergence and the Chernoff-Stein lemma bound to determine the efficacy of their tests. This chapter considers the information provided over an entire watermark instead of an individual chip. Moreover, because of the mathematical derivations herein, the statistic distributions can be directly computed, aiding security arguments.

In Section 5.1, I discuss how to manipulate the chips from cryptographic information from KDF to manipulate ranging codes to construct a watermark. Fig. 5.1 provides an intuitive statistic to determine authenticity: the convolution peak should increase. In Section 5.2, I discuss statistical measures and signal processing for determining authenticity. Next, I discuss how to derive the distributions of these statistics in the presence of adversaries, enabling the calculation of authentication security. Section 5.3 covers the non-SCER adversary, and Section 5.4 covers the SCER adversary. Finally, I apply these methods in detail to a Satellite-based Augmentation System (SBAS) (and briefly to other GNSS systems) in Section 5.5.

Because of the amount of notation in this chapter, I provide Table 5.1 for the readers convenience.

## 5.1 Combinatorial Watermark Functions

This section defines the Combinatorial Watermarking Function, which must utilize the numerical constructions from Sections 5.1.3 and 5.1.4. Section 5.1.1 discusses Chimera's watermark, the only prior art available on watermark construction. Section 5.1.2 discusses what makes a good watermarking function and compares Chimera's and the Combinatorial Watermarks.

To define the Combinatorial Watermarking Function, let $n$ be the number of chips or chip sections in a section of ranging code that will receive its own watermark. For instance, this could be $n = 1023$ for Global Positioning System (GPS) C/A signals or $n = 10230$ for L1C signals. Or this could be $n = 1023$ 10-chip sections per L1C ranging code [76]. For the Combinatorial Watermarking Function, among the $n$,

| **Notation** | **Definition and Short Description** |
|---|---|
| $n$ | Number of chips over a single watermark (i.e., the number of chips in a ranging code or section or group of ranging codes). For example, with SBAS, $n = 1023$. |
| $r$ | Number of chips inverted in a single watermark. $r$ can vary to meet specific design concerns. In [6], they suggest $r = 15$. In this chapter, I suggest $r = 4$. |
| $W$ | Number of individual watermarks considered together for a single authentication determination. In this chapter, I suggest $W = 6000$. |
| $R$ | Replica of the original, unwatermarked ranging code. |
| $R^w$ | Replica of the authentic watermarked ranging code. |
| $R^{\neg\text{SCER}}$ | Replica elected by the Non-SCER spoofer. |
| $R^{\text{HDSCER}}$ | Replica elected by the HDSCER spoofer. |
| $R_-^{(\cdot)}$ | Replica reversed in time for use in a matching convolution filter. |
| $F$ | Sampling rate of the radio involved with the authentication determination. |
| $T$ | Coherent integration time of a single watermark measurement. |
| $S$ | Signal measured over $T$ with sampling rate $F$ immediately proceeding correlation. |
| $P$ | Power of the signal within a receiver radio immediately proceeding correlation. |
| $\sigma^2$ | Noise power within a receiver radio immediately proceeding correlation. |
| $\mathcal{N}$ | Normal distribution. |
| $Y_\Delta$, $Y_\Sigma$ | Receiver statistical filters utilized for authentication determination and their random variables. |
| $s$ | Number of chips a Non-SCER adversary may elect to invert when attempting to spoof a receiver. |
| $H$ | Hypergeometric random variable relevant to the Non-SCER adversarial defense. |
| $\mathcal{H}(n, r, s)$ | Hypergeometric distribution. A Non-SCER adversary engaged in a spoofing attack generating false signals with $s$ randomly selected chips inverted will guess $H \sim \mathcal{H}(n, r, s)$ correctly for each watermark. |
| $\alpha$ | Chip estimation decision boundary elected by the adversary when utilizing a BPSK model during chip estimation of the signal. |
| $\mathcal{B}(r, p)$ | The binomial distribution with $r$ draws with draw success probability $p$. |
| $p_{\text{e}\vert r}$ | Adversary's error probability when estimating an inverted chip. |
| $p_{\text{e}\vert\neg r}$ | Adversary's error probability when estimating a noninverted chip. |
| $B_r$ | Binomial random variable of the inverted chip estimation successes. |
| $B_{\neg r}$ | Binomial random variable of the noninverted chip estimation failures. |
| $g_{\Delta,\neg\text{SCER}}$, $g_{\Sigma,\neg\text{SCER}}$ | Linear functions that transform the support of $H$ to $Y$ for mathematical conciseness, conditioned on the scheme parameters. |
| $g_{\Delta,\text{HDSCER}}$, $g_{\Sigma,\text{HDSCER}}$ | Linear functions that transform the support of $B$ to $Y$ for mathematical conciseness, conditioned on the scheme parameters. |

Table 5.1: Consolidated Combinatorial Watermark Notation Table.

GNSS will select ***exactly*** $r$ pseudorandomly to ***invert***. For maximum cryptographic entropy, the selection of $r$ must be ***uniform*** and ***one-way***, Sections 5.1.3 and 5.1.4, respectively.

The pseudorandomness can derive with KDF via the TESLA framework, as is the case for all pseudorandom information in Chapter 3 and depicted in Fig. 5.2. Fig. 5.2 depicts how GNSS can derive as much watermark pseudorandom material via KDF as needed. The depicted KDF can be used for any context. The context string "watermark" ensures no relation with other navigation message authentication (NMA) commitment-MAC function (CMF) keys. Time $t$ allows for differentiation for any watermark cadence (e.g., per 1/10th ranging code, per ranging code, per 10 ranging codes). The pseudorandom noise code assignment (PRN) allows for all satellites to derive from the same hash point (and so on with frequency and other contexts). Fig. 5.2 also enables all of the pseudorandom material to be derived in parallel to each hash point.

Within the magnifying glass of Fig. 5.2, the conceptual diagram depicts the effect of the Combinatorial Watermarking Function. Since the watermark therein derives from the **green** hash point, the $r$ inverted chips are depicted as **green** lines. In the case of Fig. 5.2, there are five inverted chips per ranging code, and each hash point watermarks several consecutive ranging codes.

To create a Combinatorial Watermarking Function, one needs two drawing algorithms capable of working with TESLA hash points. First, the construction must draw a fixed-number combination from a set uniformly. This is the subject of Section 5.1.3. The methods from Section 5.1.3 must draw one-way pseudorandom integers from the KDF-derived cryptographic pseudorandom. This is the subject of Section 5.1.4. With Sections 5.1.3 and 5.1.4, a Combinatorial Watermarking Function is uniform and one-way, so it will admit no efficient algorithm to guess TESLA hash points presumed secret and held by GNSS for the delayed disclosure time $\Theta$.

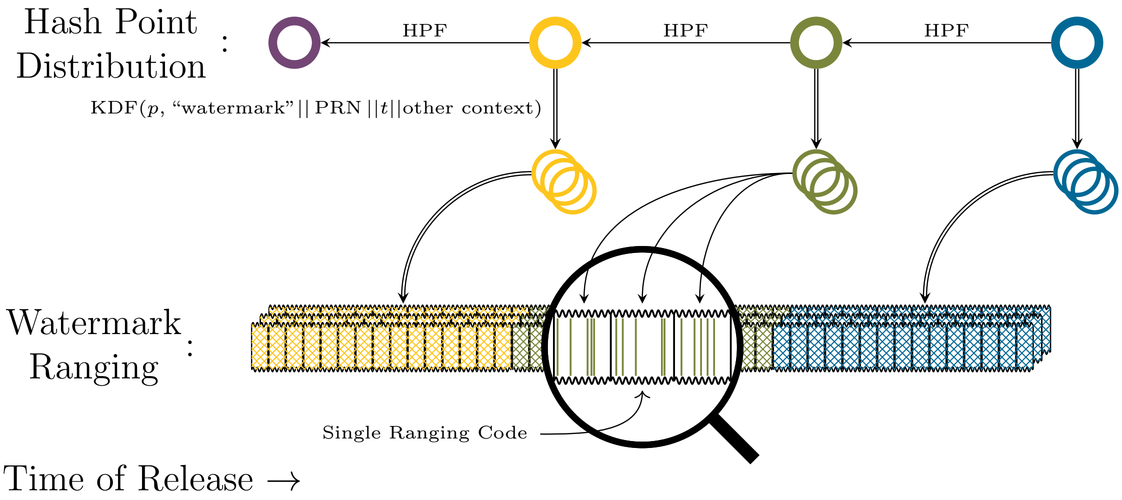

Hash Point Distribution :
HPF
HPF
HPF
KDF($p$, "watermark" || PRN ||$t$||other context)
Watermark Ranging :
Single Ranging Code
Time of Release →

Figure 5.2: Combinatorial Watermarks And KDF.

The figure depcits the crypgraphic relationship between the inverted chips, KDF, and TESLA hash points. The GNSS designer can create any context needed for KDF to construct a watermark with arbitrary parameters (e.g., $n$, $r$, number of individual watermarks per ranging code, number of watermark ranging codes per hash point). Provided the KDF context is unique, all of the pseudorandom material derived will be cryptographically independent and therefore suited for cryptographic pseudorandom functions. The pseudorandom functions that derive which $r$ chips are inverted among the $n$ are discussed in Sections 5.1.3 and 5.1.4.

### 5.1.1 Chimera's Watermark

The very first watermarking signal authentication described here is Chimera's proposal for GPS's L1C [1]. Chimera proposes two independent watermarks: Fast and Slow. The designation relates to the time to authentication (TTA). The Fast watermark serves connected receivers, and the Slow watermark serves disconnected receivers. The Fast watermark authenticates over a shorter interval since its users can retrieve the watermark over an internet connection. The Slow watermark authenticates a longer interval whose cadence is slower to align its delayed distribution with the data bandwidth availability of the L1C data channel. In Section 3.3.1, I discuss combining the two watermarks to limit the L1C Pilot watermark degradation required.

The watermark is implemented on the 10230 L1C Pilot Weil ranging codes [88]. A single-ranging code is transmitted over a 10 ms interval. The 10230-bit ranging code is divided into ten sections. Within each section, there are 1023 ranging chips. Each section is further divided into 31 sectors of 33 chips each. Among the 31 sectors, 16 are allocated to the Slow Watermark, and 15 are allocated to the Fast Watermark.

Chimera specifies a watermark duty cycle that can be adjusted. The higher the duty cycle, the more sectors (among the 16 for Slow and the 15 for Fast) will be randomly selected using Advanced Encryption Standard (AES) as a PRF. If a sector is selected to be watermarked by AES, the Weil code itself for that sector will be replaced by an AES pseudorandom sequence.

There are three aspects of pseudorandomness in this process. First, the 31 sectors are pseudorandomly divided into 16 Slow and 15 Fast sectors. While the division of the 16 and 15 appears random, it is deterministically a function of the satellite PRN number, meaning it does not change over time per satellite. The remaining two aspects are generated from AES from the Elliptic Curve Digital Signature Algorithm (ECDSA) signature for L1C's NMA. Second, the particular sector(s) are selected among the 16 and 15 (with the duty factor determining the number selected). Third is the selected replacement ranging code sequence for each chosen sector(s). While the Chimera watermark will undoubtedly achieve its intention of making spoofing attacks more difficult, how much more difficult?

To answer that question, to begin, the receiver will perform a noisy convolution of the signal with the replica. For each watermark, measurements at the sampling frequency will be added to a single number that will be compared to a threshold. Suppose the adversary spoofs a Chimera watermarked L1C ranging code with a random cryptographic initialization as a thought experiment. Because of the multiple steps in the Chimera watermark construction, it is unclear how to model the distribution of this threshold statistic decision rigorously. Therefore, Chimera cannot rigorously answer security and design questions except with Monte Carlo methods and experiments. For instance, what is the probability of missed detection (PMD) (or the security level) of a particular watermark? Or, what is the minimum duty cycle (and degradation) needed to achieve a specific PMD?

There are also weaknesses in the Chimera watermark. First, the Chimera watermark is time-correlated. Since the sector pattern is fixed per satellite, the spoofer knows the possible watermark perturbation locations (narrowed down from 31 sectors to 16 or 15 per satellite). Therefore, the adversary could observe and intentionally jam those sectors more easily. Since sectors are replaced with ciphertext, rather than directly inverting chips within the ranging code, the watermark locations are easier to determine with radio equipment via the burst-error-like behavior. The adversary could intentionally jam those sectors where it can observe replacements with AES-random sequences.

Second, the Chimera loses about half of its discrimination power. The Chimera watermark replaces segments of one pseudorandom ranging code (the Weil code) with AES-generated pseudorandom sequences. Consider the effect within the signal-replica convolution. Half of the chips will remain unchanged in expectation, depending on how the original and replacement codes align. With about half of the chips not changing, the statistical difference (intuitively depicted in Fig. 5.1) will be half compared to the number of chips manipulated and cryptographic operations.

Overall, the Combinatorial Watermark will address three themes of improvement. First, the receiver authentication statistic (whatever that statistic is) induced by the Chimera watermark is hard to model and portably apply to other GNSS signals, obstructing security analysis. Second, the watermark is time-correlated, revealing additional information, making Security Code Estimation and Replay (SCER) attacks easier. Third, the watermark loses some discrimination power due to its replacement-based construction. These considerations motivate a search for other watermarking functions, such as the Combinatorial Watermark.

### 5.1.2 Good Watermarking Functions

In this section, I discuss what makes a good watermarking function and why Combinatorial Watermarking Functions meet these characteristics. A good watermarking function

(1) reveals no information about other hidden-commitment-TESLA information,

(2) is unbiased and not time correlated, ensuring that measurement of individual chips reveals no information about other chips,

(3) has portable security analysis and induces a tractable distribution for authentication security statistics,

(4) has a constant degradation, and

(5) has each cryptographic perturbation affect the receiver authentication statistic.

### Hides Information

The observation of the watermark should not reveal other NMA CMF keys. For instance, encoding the hash point directly into the watermark would not be a good idea since this could aid the adversary with utilizing its observation to break the security of other authenticated objects within the TESLA framework. To achieve this property, Combinatorial Watermarking ensures that the underlying random drawing derives from a one-way function, as discussed in Sections 5.1.3 and 5.1.4. By ensuring that the watermarking function is one-way and cryptographically independent, the watermark reveals little information about other aspects of the authentication protocol.

### Unbiased and Not Time-Correlation

The watermarking function should be unbiased in selecting all possible watermarks derived from KDF. Then, the adversary must perform an exhaustive search to break the security of the watermark. If the function is biased, the security level of the watermark will decrease to that bias amount because the adversary can leverage that bias to create an efficient algorithm better than an exhaustive search. Sections 5.1.3 and 5.1.4 describe the underlying functions of the Combinatorial Watermarking that ensure the watermark is unbiased.

Because the watermark can be constructed to modify ranging code uniformly pseudorandomly at the chip level, the Combinatorial Watermark is not time-correlated. Not being time-correlated means that measuring one section of the watermark does

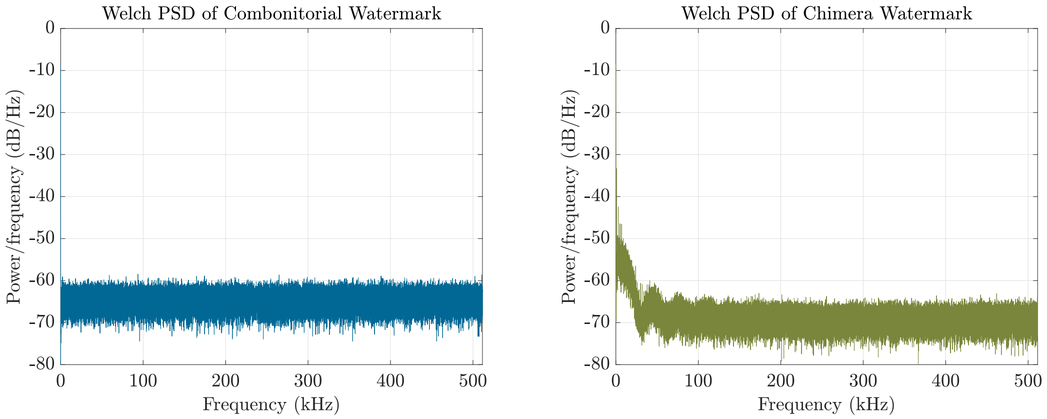


Figure 5.3: Welch PSD Comparison For Combinatorial And Chimera Watermarks.

not provide information about other sections of the watermark. Following typical cryptographic security conditions, the watermark should be indistinguishable from a random sequence when understood as an overlaid signal.

Because the Chimera Watermark first pseudorandomly selects a section and then replaces it with ciphertext, it behaves as a time-correlated burst. While observing a Chimera-watermarked ranging code, an SCER adversary can more easily detect the watermarked sections within the ranging code because the adversary knows the perturbations come in bursts. To observe the Combinatorial Watermark's improvement for time correlation, let's look at the Welch power spectral density (PSD) of the two watermarks.

Fig. 5.3 compares the Welch PSD after removing the ranging code for simulated Combinatorial and Chimera watermarks. To generate Fig. 5.3, I concatenated 1000 independent Chimera and Combinatorial watermarks after removing the ranging code. A perfect watermark will be indistinguishable from a random sequence; therefore, it should be perfectly uncorrelated and have a uniform spectrum. The Welch PSD plots show the spectra of the two, of which the Combinatorial Watermark shows a better lack of time correlation than Chimera's watermark.

### Portable Security Analysis and Tractable Distributions

Portable security analysis means the authentication security analysis need not be customized to the watermarking protocol. The analysis of one watermark applies to another watermark.

For Combinatorial Watermarking, the security analysis is described in detail in Sections 5.3 and 5.4. Throughout that analysis, everything is parameterized by $n$ and $r$. $n$ and $r$ can be any numbers with no restrictions based on the factors of the chip count. Compare to this Chimera's watermark where 1023-bit sections are divided into $31 \cdot 33 = 1023$ sections.

As discussed in Sections 5.3.1 and 5.4.3, the distribution statistics induced by Combinatorial Watermarking Functions can be computed. This means the authentication security analysis does not rely on simulated results and assumptions related to the Central Limit Theorem (CLT). The math enables direct computation of PMDs and design optimization, such as finding the minimum number of chips inverted for a specific security level.

### Constant Degradation

Constant degradation means that the amount of degradation in the authentication statistic is constant, not pseudorandom. This aids with authentication security modeling. The correlation degradation is precisely $2 \cdot \frac{r}{n}$ for Combinatorial Watermarking. Since the Chimera watermark replaces sections with AES sequences, the amount of degradation is pseudorandom. About half of the replaced chips will be aligned with the original ranging code, and about half will induce the needed degradation for authentication.

### Efficient Use of Cryptography

Finally, every cryptographic operation should affect the receiver authentication security statistic rather than having a chance of affecting the authentication security statistic. With a Combinatorial Watermark, each pseudorandom draw enacts a chip inversion, and each inversion enacts a degradation. Compare this to the Chimera watermark, where an AES operation generating a sequence will enact a degradation for about half of the chips. Reasonably, there should be a way to construct the Chimera watermark with half of the AES operations as presently described.

### 5.1.3 Pseudorandom Combination Selection

Suppose there are $n$ chip sections, and one desires to randomly select $r$ of them to invert to compose a watermark. There are several ways to draw exactly $r$ from $n$. This section specifies two. Both presume the existence of a one-way cryptographic pseudorandom integer function, for which one is provided in Section 5.1.4.

First, the GNSS provider could draw among the $n$ until it has $r$ unique drawn elements. Because there is a probability that the GNSS will draw the same element multiple times, each time a duplicate is drawn, the drawing must reject the duplicate until there are $r$ unique drawn elements. In a watermark, $r/n$ will be so small that these duplicative draws will be infrequent. To derive the expected number of draws, one must add the successive expectations of drawing a new element:

$$\mathbb{E}[\#\text{ draws}] = \sum_{i=0}^{r-1} \frac{2^{\lceil \log_2 n \rceil}}{n-i} \; . \tag{5.1}$$

The numerator's smallest base power of 2 greater than $n$ results from how a cryptographic pseudorandom generator produces strings of 0s and 1s. Groups of bits will be combined to produce a random integer between 0 and $2^{\lceil \log_2 n \rceil}$. In addition to rejecting duplicates, the drawing must reject a drawing greater than $n$ but less than $2^{\lceil \log_2 n \rceil}$.

Second, the GNSS could use a drawing proposed by Floyd [22] and provided in Algorithm 5.1. By using Algorithm 5.1, the issue of rejecting duplicates and integers from $n + 1$ to $2^{\lceil \log_2 n \rceil}$ is abstracted into a random-integer function. The issue of rejecting drawings is still there under the hood; however, Algorithm 5.1 enables the implementer to not think about it. Floyd's drawing algorithm will initiate an exact number of calls to the random integer function. I provide a short proof of correctness

**Algorithm 5.1:** A Function That Selects A Pseudorandom Combination Of Integers Uniformly From A Pseudorandom Integer Function.

```
/* n:  The total number of elements that can form any
   combination.                                              */
/* r:  The number of elements selected for a particular
   combination.                                              */
/* seed:  The pseudorandom seed from which the uniform
   combination will derive.                                  */
function pseudorandom_combination(n, r, seed)
    combination = Set() /* Initialize set data structure      */
    kdf_rand_int_cache /* Initialize empty kdf_kdf_int cache.  */
    for j = n-r+1, ..., n do
        /* Derive random integer from 1 to j in a cryptographic
           one-way manner.                                   */
        rand_int, kdf_rand_int_cache = kdf_rand_int(1, j, seed,
         *kdf_rand_int_cache)
        /* Make unbiased selection.                          */
        if rand_int not in combination then
            combination.add(rand_int)
        else
            combination.add(j)
        end
    end
    return combination
```

in the following subsection.

In addition, the issue of efficiently using cryptographically generated pseudorandom material is abstracted to the random integer function. For instance, suppose the underlying randomness derives from keyed-hash message authentication code (HMAC)-SHA256. Each HMAC-SHA256 call produces 256 bits of pseudorandom material, which can be used for multiple random drawings rather than a single integer.

### Proof of Floyd's Drawing

For the reader's convenience, here I provide a proof that Algorithm 5.1 selects a combination where the probability of the selection of any element is the same, inspired by [22, 26]. To begin, suppose there are $n$ total elements from which one will draw a uniform combination of $r$ elements. Within Algorithm 5.1, the for-loop iterates $j$ from $n-r+1$ to $n$ (inclusive). Let $\mathrm{Pr}_j(\neg i)$ be the probability that element $i$ is not selected with the $j$-th for-loop iteration, and let $\mathrm{Prob}(\neg i)$ be the probability that element $i$ was never selected after the last iteration of the for-loop. I will show that $\mathrm{Pr}(\neg i)$ is the same for all $i$. To do this, I must split $\mathrm{Pr}(\neg i)$ into two cases that correspond to whether the if conditional within Algorithm 5.1 could activate: (A) $1 \leq i \leq n-r+1$ and (B) $n-r+1 < i \leq n$.

For Case A, I start with the very first draw, corresponding to when $j = n-r+1$, $P_{n-r+1}(\neg i) = \frac{n-r}{n-r+1}$. Thereafter, $\mathrm{Pr}_j(\neg i) = \frac{j-1}{j}$. Therefore,

$$\begin{aligned}\mathrm{Pr}(\neg i) &= \prod_{j=n-r+1}^{n} \mathrm{Pr}_j(\neg i)\\ &= \frac{n-r}{n-r+1} \cdot \frac{n-r+1}{n-r+2} \cdots \frac{n-1}{n}\ ;\\ \mathrm{Pr}(\neg i) &= \frac{n-r}{n}\ .\end{aligned}$$

For Case B, $i$ is not in the range of the random integer drawn until $j = i$. Therefore, $\mathrm{Pr}_j(\neg i) = 1 \quad \forall j < i$. On the $j = i$ draw, following through the for-loop iteration, either the $i$-th element is selected via the random integer function or the $i$-th element is selected because the random integer function provided an element previously drawn. Among the $i$ possible elements to draw, $i-(n-r+1)+1$ outcomes lead to drawing $i$, and $n-r$ outcomes lead to not drawing i. Therefore, $P_i(\neg i) = \frac{n-r}{i}$.

Thereafter, like with Case A, $\Pr_j(\neg i) = \frac{j-1}{j}$. Therefore,

$$\begin{aligned}
\Pr(\neg i) &= \prod_{j=n-r+1}^{n} \Pr_j(\neg i) \\
&= \prod_{j=i}^{n} \Pr_j(\neg i) \\
&= \frac{n-r}{i} \cdot \frac{i}{i+1} \frac{i+1}{i+2} \cdots \frac{n-1}{n} \; ; \\
\Pr(\neg i) &= \frac{n-r}{n} \; .
\end{aligned}$$

Since both Case A and Case B yield the same $\Pr(\neg i)$, each element $i$ is selected uniformly among the combinations of $r$ elements.

### 5.1.4 One-way Pseudorandom Integers

Section 5.1.3 requires a cryptographic pseudorandom integer function. This section discusses such a function based on KDF with Algorithm 5.2. Other pseudorandom integer functions could provide the random integers required for Algorithm 5.1; however, Algorithm 5.2 meets several desired properties relevant to security.

One such property is that Algorithm 5.2 is a one-way cryptographic function. This property will ensure that knowledge provides no information about the generating seed. Because Algorithm 5.2 utilizes KDF to derive the integers, it has the one-way property. Algorithm 5.2 admits no efficient algorithm to determine the input seed from the output integers.

An adversary can observe the watermark with a radio. Upon observing an inverted chip section in the ranging code, the adversary can deduce the integers derived from the watermarking function. By ensuring that the integer derives from a one-way function, the adversary is limited in predicting the watermarking seed and future inverted chip sections in the remainder of the watermark. This also allows the integers in Fig. 5.1 and similar figures to come from a one-way arrow.

Another desired property is efficiency with computational resources. Suppose the KDF function is HMAC-SHA256, which produces 256 bits per call. One naive way to

**Algorithm 5.2:** A One-way Random Integer Function Based On KDF.

```
/* a:  The minimum output integer.                                */
/* b:  The maximum output integer.                                */
/* seed:  Input to KDF key field for generating random bits.      */
/* counter:  KDF message field input to ensure non-periodicity
   of KDF pseudorandom output.                                    */
/* start_bit:  The first bit of kdf cache not yet used to
   generate a random integer.                                     */
/* kdf_cache:  Cache of random bits from the last KDF call.       */
function kdf_rand_int(a, b, seed, counter=0, start_bit=0,
 kdf_cache="")
    /* Generate new pseudorandom bits for the cache.              */
    if kdf_cache=="" then
        kdf_cache = KDF(seed, counter)
    end
    /* KDF Output Length (e.g., 256 for HMAC-SHA256)              */
    max_bit = KDF.length
    /* Compute needed number of bits to commence random draw      */
    bits_needed = FLOOR(LOG2(b-a))+1
    /* If not enough unused bits of KDF cache, then recursively
       redraw with updated counter                                */
    if bits_needed > max_bit - start_bit then
        return kdf_rand_int(a, b, seed, counter+1, 0, "")
    end
    /* Use some random bits and cast as integer                   */
    random_int = INT(kdf_cache[start_bit:start_bit + bits_needed])
    if random_int > (b-a) then
        /* Reject and redraw integer if outside desired range     */
        return kdf_rand_int(a, b, seed, counter, start_bit+bits_needed,
         kdf_cache)
    else
        /* Return unbiased one-way random integer and cache
           required to generate additional random integers at next
           kdf_rand_int call.                                     */
        return a + random_int, (counter, start_bit+bits_needed, kdf_cache)
    end
```

derive the integers required is to produce an HMAC for each derived integer. HMAC allows the pseudorandomness to be derived in parallel (as opposed to via repeated hash application). Unfortunately, there is no way to sample an arbitrary random integer with a deterministic runtime unless the range of integers is a power of 2. For an arbitrary random integer range, the drawing must sample with rejection to ensure that the pseudorandom integer function output is not biased. For instance, the output of HMAC cast to be an integer can be rejected if outside the desired integer range in favor of new HMACs until the output is within the desired range. If the output of HMAC is 256 bits, but the range might be considerably smaller (e.g., 1 to 1023), the function would reject most outputs. To derive an integer $i$, one should use the smallest number of bits required to delineate the range of positive integers up to the smallest power of 2 larger than $i$. Therefore, it is desirable to sparingly use the output bits from HMAC so that multiple random integers can be derived from a single call to HMAC.

Algorithm 5.2 has all the above properties. Because the random bits used to generate the random integers derive from a KDF derived from the inputs with Algorithm 5.2, the function satisfies the one-way property. The function caches the output of KDF to ensure the minimum number of KDF calls. Algorithm 5.2 ensures the minimum required number of bits is used for generating each integer. The function uses a cache containing a counter and start bit to ensure non-periodicity with Algorithm 5.2. The recursive structure achieves sample rejection with a concise program.

## 5.2 Signal Processing

To enact watermark ranging authentication, the receiver must process the pseudoranges twice, as depicted in Fig. 5.1: once before knowledge of the watermark and once after knowledge of the watermark. The receiver must store the ***raw, intermediate frequency baseband*** samples to complete the second processing. This represents a change to the required receiver hardware. The receiver cannot accumulate and dump because the receiver must recall the samples later for the second processing step. This amounts to at least the Nyquist Frequency across the entire TESLA ***interval***, which might require specialized hardware such as a ring buffer.

When determining the authenticity of a signal, a receiver must map relevant information from the raw intermediate frequency radio data to a statistic against a threshold. For the statistic to be useful, the PMD and the probability of false alarm (PFA) given the selected threshold should be small. The statistics from the literature usually suggest something related to the correlations between the data and watermarked and unwatermarked replicas and make security arguments based on ***simulated*** likelihoods. The security arguments of this chapter will be based on ***derived*** probability distributions.

This section describes two statistics the receiver must compute and compare against a threshold for the second processing step. Section 5.2.1 defines them; however, my selection of specific mathematical details (e.g., the gain coefficients) will become apparent in Sections 5.2.2, 5.3.1 and 5.4.3. The advantage in selecting the two statistics herein (rather than a single one) will not be apparent until Section 5.4.3. Section 5.2.2 derives the distribution of those statistics when there is no spoofer. The derivations of this derive the distribution of those statistics when there are spoofers. Sections 5.3 and 5.4 discuss the cases when there are spoofers. Throughout the derivations of this chapter, many equations rely on an estimate of the signal power and noise, which is the subject of Section 5.2.3. And Section 5.2.4 briefly describes an approach to model the effect of the receiver measurement quantization.

### 5.2.1 Receiver Statistics

For the security arguments presented in Sections 5.3 and 5.4, the receiver should utilize the statistics of this section. While these statistics pose conveniences that are apparent in later sections, there could be other statistics. Fig. 5.4 provides a block diagram of the filters needed to generate the two statistics.

The two statistics of this section will derive from two linear time-invariant (LTI) filters where the receiver samples evenly with frequency $F$ over a coherent integration $T$. Let $R \in \{-1, 1\}^{FT}$ be a replica of the ***unmodified*** ranging code for examination. $R$ would be a ***resampled*** Gold Code if it were GPS C/A, or it could be sections of

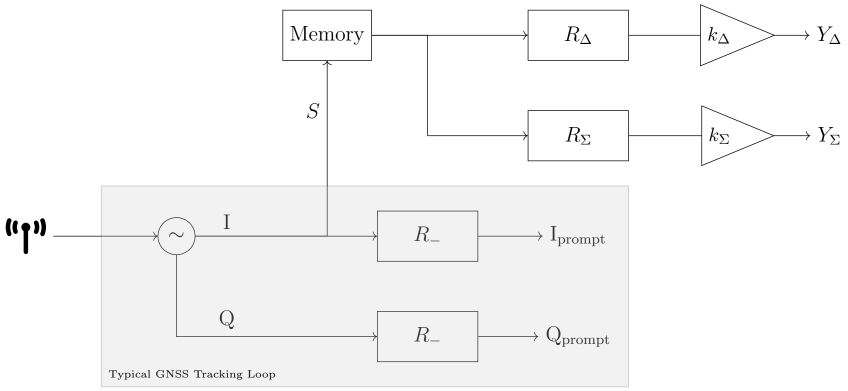

Memory
$R_\Delta$
$k_\Delta$
$Y_\Delta$
$S$
$R_\Sigma$
$k_\Sigma$
$Y_\Sigma$
~
I
$R_-$
$\mathrm{I}_{\mathrm{prompt}}$
Q
$R_-$
$\mathrm{Q}_{\mathrm{prompt}}$
Typical GNSS Tracking Loop

Figure 5.4: Combinatorial Watermarking Receiver Signal Processing Diagram.

The block diagram represents how the receiver will need to process the signal to assert authentication. Depicted is the typical tracking loop for a receiver tracking GPS's C/A signal, where the signal is in phase, as an example. The receiver must store into memory to await distribution of the hash point the raw baseband samples of the watermarking ranging code (i.e., before correlation) within a converged tracking loop. Thereafter, the receiver must correlate with two replicas, $R_\Delta$ and $R_\Sigma$, to generate statistics $Y_\Delta$ and $Y_\Sigma$. The receiver then makes an authentication determination based on thresholds of $Y_\Delta$ and $Y_\Sigma$.

ranging code. However, $R$ will cover a single watermark, meaning it derives from the original ranging code containing $n$ chips (or chip sections). This means that $||R||_1 = ||R||_2^2 = FT$.

The authentic watermarked signal will have $r$ inverted among the $n$. Let $R^w \in \{-1, 1\}^{FT}$ be the watermarked replica. $R^w$ is not known to anyone except the GNSS provider until after the distribution of the hash point. As depicted in Fig. 5.4, the receiver assembles the two LTI filters:

$$R_\Delta = R_-^w - R_- \ ; \tag{5.2}$$

$$R_\Sigma = R_-^w + R_- \ . \tag{5.3}$$

The statistic is just a simple correlation of the signal with these replicas. However, noting that these are ***valid*** convolutions with an LTI filter will aid with propagating noise analysis later. The replicas $R_\Delta$ and $R_\Sigma$ are the impulse responses of the filters. And because these are LTI filters producing matching convolutions, each of the replicas has a $-$ subscript to note that the filter convolves the ***reversed-in-time*** replica.

The gains are selected for mathematical convenience, which will be apparent later. Let $P$ be the power of the signal, whose estimation process is discussed in Section 5.2.3. The gains from Fig. 5.4 are the following:

$$k_\Delta = \frac{1}{||R^w - R||_1}\frac{1}{\sqrt{P}} = \frac{1}{2r}\frac{n}{FT}\frac{1}{\sqrt{P}}\ ; \tag{5.4}$$

$$k_\Sigma = \frac{1}{||R^w + R||_1}\frac{1}{\sqrt{P}} = \frac{1}{2(n-r)}\frac{n}{FT}\frac{1}{\sqrt{P}}\ . \tag{5.5}$$

In Eqs. (5.4) and (5.5), the second equality (i.e., with the $\frac{n}{FT}$ fraction) presumes that the sample rate is evenly distributed and that the effect of the watermark is uniform across the coherent integration. $FT$ will not be a multiple of $n$ to avoid signal processing artifacts; however, the uniform effect is provided by Section 5.1.3.

A digital receiver will convert the signal analog measurement into quantization levels, and a gain stage is applied prior to this conversion to ensure the digital representation avoids saturation. For this work, I will avoid these effects by assuming that the signal can be any real number for mathematical conciseness purposes. In Section 5.2.4, I describe a suggested approach to repeat the analysis of this chapter to account for this effect. But for now, suppose the signal measured by the receiver over the interval $T$ after removing the carrier wave is $S \in \mathbb{R}^{FT}$. $S$ could be spoofed or not.

To model $S$, I split it into two components: the ranging code with a signal power and additive white Gaussian noise (AWGN) $N \in \mathbb{R}^{FT}$ with $N \sim \mathcal{N}(0, \sigma^2 I)$ and $\sigma^2$ the noise power (i.e., $\mathbb{E}[NN^\top] = \sigma^2 I$):

$$S^{\text{auth}} = \sqrt{P}R^w + N\ . \tag{5.6}$$

I note that the noise power can be tricky to determine (e.g., in the GPS C/A case, the noise power would be half the total noise since only the in-phase portion contributes in Fig. 5.4). The distributions of $Y_\Delta$ and $Y_\Sigma$ with signal $S$ in the authentic case are derived in Section 5.2.2. When $S$ is spoofed, the signal is composed of a different $R$

(unless the adversary's guess was lucky). How the adversary selects its own spoofed $R$ and the effects on $Y_\Delta$ and $Y_\Sigma$ are the subjects of Sections 5.3 and 5.4.

Moreover, when $S$ is spoofed, the noise could be something else that is not AWGN. However, given the noise characteristics of GNSS receivers, the receiver will need to observe many watermarks before making an authentication determination. Because the receiver will average a large number of $Y_\Delta$ and $Y_\Sigma$ to make an authentication determination, the effect of non-AWGN noises will diminish due to the CLT.

#### Statistics Intuition

For intuition and mathematical conciseness, suppose that $FT = n$. Therefore $R, R^w \in \{-1, 1\}^n$ and there are exactly $r$ elements in $R^w$ that are the negative of $R$ and the other $n - r$ are identical.

$R_\Delta$ is the subtraction of the watermarked replica with the unwatermarked replica. Because the watermark is a small perturbation, this subtraction will mostly be 0. Exactly $r$ elements of $R_\Delta$ will be either 2 or $-2$, depending on where the ranging code was 1 or $-1$ before inversion. This means that the $Y_\Delta$ filter discards all of the signal information except where the inverted chips should be. $Y_\Delta$ measures how well the adversary can predict watermarked chips.

$R_\Sigma$ is the sum of the watermarked replica and the unwatermarked replica. Because the watermark is a small perturbation, the addition will mostly be 2 or $-2$, depending on where the ranging code was 1 or $-1$ before inversion. Exactly $r$ elements of $R_\Sigma$ will be 0. This means that the $Y_\Sigma$ filter discards all of the watermark information. $Y_\Sigma$ measures how well the receiver will track the signal in the presence of the watermark.

### 5.2.2 The Authentic Case

To derive the distribution of each $Y$ under authentic conditions, one must propagate $S$ through the LTI filters. With the knowledge that the statistics $Y_\Delta$ and $Y_\Sigma$ derive from LTI filters, the derivation involves convolutions and noise propagation:

$$\begin{aligned}
Y_\Delta &= k_\Delta \cdot R_\Delta * S^{\text{auth}} \\
&= k_\Delta \cdot (R_-^w - R_-) * (\sqrt{P} R^w + N) \\
&= k_\Delta \sqrt{P} \cdot (R_-^w * R^w - R_- * R^w) + k_\Delta \cdot (R_-^w - R_-) * N \\
&= k_\Delta \sqrt{P} \cdot \left( FT - \frac{n-2r}{n} FT \right) + k_\Delta \cdot (R_-^w - R_-) * N \\
&= k_\Delta \sqrt{P} \cdot 2r \frac{FT}{n} + k_\Delta \cdot (R_-^w - R_-) * N \; ; \\
Y_\Delta &= 1 + k_\Delta \cdot (R_-^w - R_-) * N \; . \qquad (5.7)
\end{aligned}$$

In the authentic case, the distribution of $Y_\Delta^{\text{auth}}$ is a shifted normal distribution where the mean is 1. Repeating for $Y_\Sigma^{\text{auth}}$ yields

$$\begin{aligned}
Y_\Sigma &= k_\Sigma \cdot R_\Sigma * S^{\text{auth}} \\
&= k_\Sigma \cdot (R_-^w + R_-) * (\sqrt{P} R^w + N) \\
&= k_\Sigma \sqrt{P} \cdot (R_-^w * R^w + R_- * R^w) + k_\Sigma \cdot (R_-^w + R_-) * N \\
&= k_\Sigma \sqrt{P} \cdot \left( FT + \frac{n-2r}{n} FT \right) + k_\Sigma \cdot (R_-^w + R_-) * N \\
&= k_\Sigma \sqrt{P} \cdot 2(n-r) \frac{FT}{n} + k_\Sigma \cdot (R_-^w + R_-) * N \; ; \\
Y_\Sigma &= 1 + k_\Sigma \cdot (R_-^w + R_-) * N \; . \qquad (5.8)
\end{aligned}$$

Given the distribution of $N$, the distributions of $Y_\Delta^{\text{auth}}$ and $Y_\Sigma^{\text{auth}}$ easily follow via noise propagation with an LTI filter. The expectations are trivially $\mathbb{E}[Y_\Delta^{\text{auth}}] = \mathbb{E}[Y_\Sigma^{\text{auth}}] = 1$. Because the distributions are the sum of a normal distribution and a constant, deriving the variances completely describe the distributions. For the $Y_\Delta$ case:

$$\begin{aligned}
\mathbb{V}[Y_\Delta^{\text{auth}}] &= \mathbb{V}[1 + k_\Delta \cdot (R_-^w - R_-) * N] \\
&= \mathbb{V}[k_\Delta \cdot (R_-^w - R_-) * N] \\
&= k_\Delta^2 \cdot \mathbb{V}[(R_-^w - R_-) * N] \\
&= k_\Delta^2 \cdot ||R_-^w - R_-||_2^2 \cdot \mathbb{V}[N] \\
&= \left(\frac{1}{2r}\frac{n}{FT}\frac{1}{\sqrt{P}}\right)^2 \cdot 4\frac{r}{n}FT \cdot \sigma^2 \ ; \\
\mathbb{V}[Y_\Delta^{\text{auth}}] &= \frac{1}{r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P} \ . \qquad (5.9)
\end{aligned}$$

And, for the $Y_\Sigma$ case:

$$\begin{aligned}
\mathbb{V}[Y_\Sigma^{\text{auth}}] &= \mathbb{V}[1 + k_\Sigma \cdot (R_-^w + R_-) * N] \\
&= \mathbb{V}[k_\Sigma \cdot (R_-^w + R_-) * N] \\
&= k_\Sigma^2 \cdot \mathbb{V}[(R_-^w + R_-) * N] \\
&= k_\Sigma^2 \cdot ||R_-^w + R_-||_2^2 \cdot \mathbb{V}[N] \\
&= \left(\frac{1}{2(n-r)}\frac{n}{FT}\frac{1}{\sqrt{P}}\right)^2 \cdot 4\frac{n-r}{n}FT \cdot \sigma^2 \ ; \\
\mathbb{V}[Y_\Sigma^{\text{auth}}] &= \frac{1}{n-r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P} \ . \qquad (5.10)
\end{aligned}$$

Therefore, combining the above, I arrive at the final distributions for the authentic case:

$$Y_\Delta^{\text{auth}} \sim \mathcal{N}\left(1, \frac{1}{r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P}\right) \ ; \qquad (5.11)$$

$$Y_\Sigma^{\text{auth}} \sim \mathcal{N}\left(1, \frac{1}{n-r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P}\right) \ . \qquad (5.12)$$

Because of the typical signal-to-noise ratio (SNR) within a GNSS receiver, the receiver will need to aggregate multiple watermarks to make a authentication determination. Suppose that the receiver averages the $Y$s over $W$ independent watermarks to make an authentication determination. Then the authentic distributions exactly become

$$Y^{\text{auth}}_{\Delta,W} \sim \mathcal{N}\left(1, \frac{1}{r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P} \cdot \frac{1}{W}\right) \ ; \tag{5.13}$$

$$Y^{\text{auth}}_{\Sigma,W} \sim \mathcal{N}\left(1, \frac{1}{n-r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P} \cdot \frac{1}{W}\right) \ . \tag{5.14}$$

Because these normal distributions will reappear in the nonauthentic cases, for notational convenience, I will abbreviate the ***zero-mean*** Gaussian components as the following:

$$N_{\Delta,W} \sim \mathcal{N}_\Delta\left(0, \frac{1}{r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P} \cdot \frac{1}{W}\right) \ ; \tag{5.15}$$

$$N_{\Sigma,W} \sim \mathcal{N}_\Sigma\left(0, \frac{1}{n-r} \cdot \frac{n}{FT} \cdot \frac{\sigma^2}{P} \cdot \frac{1}{W}\right) \ ; \tag{5.16}$$

$$Y^{\text{auth}}_{\Delta,W} = 1 + N_{\Delta,W} \ ; \tag{5.17}$$

$$Y^{\text{auth}}_{\Sigma,W} = 1 + N_{\Sigma,W} \ . \tag{5.18}$$

### 5.2.3 Power and Noise Estimation

The distributions in Sections 5.2.2, 5.3 and 5.4 presume the receiver knows the signal power $P$ and the ambient noise $\sigma^2$, pre-correlation. Whereas, in the ranging correlation problem from Fig. 1.1, the receiver need only know where the peak signal power occurs; in the authentication case, the receiver must predict these peak values. The statistics from Section 5.2.1 normalize against $P$ and $\sigma^2$, producing values centered at 1 in the authentication case. However, the adversary could manipulate $P$ and $\sigma^2$ estimators.

In this chapter, I will perform several experiments on theoretical watermark schemes to show the efficacy of the distributions derived in this chapter. Each experiment involves the GPS or Wide Area Augmentation System (WAAS) C/A code. These signals involve a quadrature phase shift key (QPSK) signal where only the in-phase portion is utilized. This means that within the tracking loop immediately preceding correlation, only half of the environmental noise is present (since the other half is in the quadrature phase). To estimate $\hat{P}$ and $\hat{\sigma}^2$, I suggest the following procedure; however, it must be modified for other signals that are not QPSK-only-use-quadrature-phase signals.

Receivers will not be sensitive enough to make an authentication determination based on a 1 ms or a 10 ms coherent integration, meaning that the decision should incorporate longer integration. This naturally leads to a simple way to regress an accurate $P$ from those individual integrations. I suggest dividing the mean prompt correlation (without knowledge of the watermark) divided by the expected gain from the correlator:

$$\widehat{\sqrt{P}} = \frac{\text{mean}(I_p)}{R_- * R^w} = \frac{\text{mean}(I_p)}{(1 - 2\frac{r}{n})FT} \ . \tag{5.19}$$

In Eq. (5.19), I use a watermarked prompt correlation, and I divide by $R_- * R^w$ because the signal would be watermarked in a real scheme.

Regressing $\sigma^2$ is trickier. But I found utilizing the existing Moments Method $\frac{C}{N_0}$ estimator post correlation from [23] and then backtracking a $\sigma^2$ utilizing derivations from [40] with Eq. (5.20) produced the best results in the experiments in this chapter:

$$\hat{\sigma}^2 = \frac{1}{2}\frac{N_0}{C}B_{\text{eq}}P||R||_1 = \frac{1}{2}\frac{N_0}{C}F\hat{P} \ . \tag{5.20}$$

In Eq. (5.20), I take the $\frac{C}{N_0}$ back through the correlation by dividing by $||R||_1$, correcting for the equivalent bandwidth $B_{\text{eq}}$, and multiplying by $\frac{1}{2}$ since the pre-correlation $S$ has been stripped of the quadrature components, leaving a noise-to-signal ratio that is multiplied by the regressed $P$. Alternative ways could be more effective, including those that directly regress or bootstrap the ratio. I defer to future work to ensure that the regression is resistant (or provably immune) to manipulation by an adversary. However, I suspect that adversarial attacks will always increase $\sigma^2$, which will be appropriately accounted for in the receiver's confidence in authenticity.

To account for the concern that the adversary could manipulate $\hat{P}$ and $\hat{\sigma}^2$ to break authentication security, I suggest designing watermarking schemes that presume low $\frac{C}{N_0}$ conditions. The presumed conditions are low enough that I expect a normal receiver to be unable to function in them. As $\frac{C}{N_0}$ approaches a reasonable level, the tests utilizing the statistics of Section 5.2.1 will then exceed the specificity and sensitivity provided by the assumption of unreasonably low $\frac{C}{N_0}$. Moreover, because the $\hat{P}$ and $\hat{\sigma}^2$ derive from averages, the adversaries jamming manipulations will suffer from the concentrating effect of the CLT. Therefore, across the domain of usable $\frac{C}{N_0}$, the spoofer should not be able to exploit manipulating $P$ and $\sigma^2$, but I disclaim that the methods of this section are provably so.

### 5.2.4 Accounting for Quantization

The incorporated distributions of this work act on support that could include all the real numbers. For instance, the normal distributions from Section 5.2.2. In Sections 5.3 and 5.4, the statistic distributions incorporate discrete distributions (e.g., hypergeometric and binomial). While they have limited support, the radio's quantization does not account for that, and so, from the radio's perspective, those distributions have discrete support among all real numbers.

Radios cannot make positive and negative infinite measurements. Instead, they make a quantized measurement over a limited domain of buckets. And presumably, the radio leverages automated gain control to ensure that its quantization limits the amount of information lost.

In Sections 5.3 and 5.4, I will use discrete probability distributions and convolution to compute the distributions of the statistics from Section 5.2.1. Because the formulations are already discrete, one can modify the formulations to account for the radio quantization. For instance, the discrete probability distributions can be aggregated into the buckets of the radio and then repeatedly convolved as described in Sections 5.3.1 and 5.4.3. Then, the same security arguments can be made about the AWGN components, albeit the nice formulations from Sections 5.3 and 5.4 will disappear. But because Sections 5.3 and 5.4 will compute probability distributions via convolution, the underlying convolutions can be changed to account for a jamming noise model and a receiver quantization.

## 5.3 Non-SCER Adversary Security

In Section 5.2.1, I describe two authentication statistics, $Y_\Delta$ and $Y_\Sigma$, that derive from LTI filters. Then, in Section 5.2.2, I derive the distributions of $Y_\Delta^{\text{auth}}$ and $Y_\Sigma^{\text{auth}}$ in the authentic case where there is no spoofer. In this section, I derive the distributions for Non-SCER adversaries. The Non-SCER does not attempt to observe the watermark, meaning this adversary will spoof by inverting chips without knowledge of the transmitted watermark.

To derive the distributions of $Y_\Delta$ and $Y_\Sigma$ under adversarial conditions, I must introduce a new variable $s$. $s$ is the number among $n$ the adversary ***elects*** to invert in its spoofed signal transmission. This means the signal processing security adversarial model must include any election $s$.

Practically, the adversary would invert $s = 0$ up to $s = \frac{n}{2}$. Because of the problem symmetry, $s > \frac{n}{2}$ is covered already by $s < \frac{n}{2}$. Inverting half of $n$ yields a new pseudorandom sequence unrelated to the original ranging code. Inverting 75% of the ranging code is the same as inverting 25% with a carrier wave phase change or a data bit flip (in the case of GPS C/A).

### 5.3.1 Statistic Distributions

Recall that an SCER adversary may attempt to observe and replay the watermark. Therefore, a Non-SCER does not observe the watermark. Because of the construction of the watermark, the adversary may only do an impossibly large exhaustive search to guess the hash point that derives the watermark. And because of the onboard GNSS-independent clock (GIC), the receiver will only accept one attempt to authenticate a particular watermarked ranging code. Therefore, the adversary may only attempt once to spoof a watermarked ranging code, and the adversary must make a guess.

The Non-SCER adversary will uniformly select $s$ from $n$ and submit a spoofed ranging code to the receiver. Depending on the thresholds for $Y_\Delta$ and $Y_\Sigma$, the best the Non-SCER adversary can do is $s = \frac{n}{2}$, corresponding to submitting a completely random ranging code to the receiver. Moreover, the spoofed signal will have constant power $P$ because the adversary does not have reason to change $P$ over a watermarked ranging code. Because of the uniformity of the watermark and because the Non-SCER adversary does not attempt to observe the watermark, there is no advantage to having the power be a function of time.

In addition, the receiver must assume that it can securely know a lower bound on the signal-to-noise ratio. In the presence of jamming, the resultant noise distribution would behave as elevated AWGN because the receiver will perform its authentication determination over many watermarked ranging codes with an average. Therefore, I will ignore the effect induced by arbitrary noises and noise distributions by the adversary.

Given $s$, the adversary will select a replica $R^{\neg\text{SCER}}$ which is exactly $s$ inversions from $R$. ¬SCER means "not SCER" because the $s$ are selected randomly without regard to any measurements of the watermark. Therefore, the spoofed signal will be

$$S^{\neg\text{SCER}} = \sqrt{P} R^{\neg\text{SCER}} + N \ . \tag{5.21}$$

Before passing the signal through the filters, it would be helpful to understand two component convolutions. First, because $R^{\neg\text{SCER}}$ is exactly $s$ inversions from $R$,

$$R_{-} * R^{\neg\text{SCER}} = (n - 2s)\frac{FT}{n} \ . \tag{5.22}$$

Second, suppose that every uniformly selected $s$ by the adversary was incorrect. Then $R^{\neg\text{SCER}}$ is both $r+s$ inversions from $R^w$: the $r$ inverted by the GNSS provider and the $s$ incorrectly inverted by the adversary. And because each inversion flips a 1 to a -1, the difference is 2 for each inversion. Therefore, $R_{-} * R^{\neg\text{SCER}}$ will be $(n-2r-2s)\frac{FT}{n}$ in the worst case. But the adversary will get some correct according to the hypergeometric distribution $\mathcal{H}(n, r, s)$:

$$H \sim \mathcal{H}(n, r, s) \tag{5.23}$$

$$\text{PMF}_H(h) = \frac{\binom{r}{h}\binom{n-r}{s-h}}{\binom{n}{s}} \tag{5.24}$$

$$R^w_{-} * R^{\neg\text{SCER}} = (n - 2r - 2s + 4H)\frac{FT}{n} \ . \tag{5.25}$$

The hypergeometric distribution describes the scenario where there are two independent, no-replacement drawings on a set $n$ and provides the probability that the two drawings pick some identical elements. The first drawing results from the GNSS provider, and the second from the adversary. If the adversary guesses $h$ correctly, the distance between $R^w$ and $R^{\neg\text{SCER}}$ shrinks by 2: one from $r$ and one from $s$. Each correct guess then increases the $R^w * R^{\neg\text{SCER}}$ convolution by 4. Therefore, I arrive at Eq. (5.25).

Now I pass $S^{\neg\text{SCER}}$ through the LTI filters from Section 5.2.1. For mathematical conciseness, let

$$y = g_{\Delta,\neg\text{SCER}}(h|r,s) = \frac{1}{2r} \cdot (4h - 2r) \ . \tag{5.26}$$

Because $g_{\Delta,\neg\text{SCER}}$ is a linear function of $h$, $g_{\Delta,\neg\text{SCER}}$ will stretch and move the support of $H$ onto the support of $Y_\Delta$. $g_{\Delta,\neg\text{SCER}}$ maps the hypergeometric domain of $H$ onto the statistic domain $Y_\Delta$.

For the $Y_\Delta^{\neg\text{SCER}}$ case:

$$\begin{aligned}
Y_\Delta^{\neg\text{SCER}} &= k_\Delta \cdot R_\Delta * S^{\neg\text{SCER}} && (5.27)\\
&= k_\Delta \cdot (R_-^w - R_-) * (\sqrt{P} R^{\neg\text{SCER}} + N)\\
&= k_\Delta \sqrt{P} \cdot (R_-^w * R^{\neg\text{SCER}} - R_- * R^{\neg\text{SCER}}) + k_\Delta \cdot (R_-^w - R_-) * N\\
&= k_\Delta \sqrt{P} \cdot (n - 2r - 2s + 4H - n + 2s)\frac{FT}{n} + k_\Delta \cdot (R_-^w - R_-) * N\\
&= k_\Delta \sqrt{P} \cdot (4H - 2r)\frac{FT}{n} + k_\Delta \cdot (R_-^w - R_-) * N\\
&= \frac{1}{2r} \cdot (4H - 2r) + k_\Delta \cdot (R_-^w - R_-) * N\\
&= g_{\Delta,\neg\text{SCER}}(H) + k_\Delta \cdot (R_-^w - R_-) * N \ ; && (5.28)\\
Y_\Delta^{\neg\text{SCER}} &= g_{\Delta,\neg\text{SCER}}(H) + N_\Delta \ ; && (5.29)\\
\text{PMF}_{Y_\Delta^{\neg\text{SCER}}}(y) &= ((\text{PMF}_H \circ g_{\Delta,\neg\text{SCER}}^{-1}) * \text{PMF}_{N_\Delta})(y) \ . && (5.30)
\end{aligned}$$

For the normal noise term in Eq. (5.28) to Eq. (5.29), I refer to Section 5.2.2. In

Eq. (5.30), the $g^{-1}$ results with how $g$ affects the domain of random variables during the linear transformation. To go from Eq. (5.29) to Eq. (5.30), recall that the PDF / PMF of the sum of two random variables is the convolution of their PDF / PMFs. In Eq. (5.30), the normal PDF is discretized into a PMF to combine with the PMF of $H$.

Suppose that a receiver will observe groups of $W$ coherent integrations with $W$ independent watermarks and average the $W$ observed $Y_\Delta$. For instance, in Section 5.5, the receiver will aggregate $W = 6000$ watermarks over 6 seconds. This average will be compared against a threshold for authentication determination. One can further convolve the distribution from Eq. (5.30). With $(\cdot)^{*W}$ meaning repeated convolution $W$ times:

$$\begin{aligned} Y_{\Delta,W}^{\neg\text{SCER}} &= \frac{1}{W}\sum_{W} g_{\Delta,\neg\text{SCER}}(H) + \frac{1}{W}\sum_{W} N_\Delta \ ; && (5.31) \\ \text{PMF}_{Y_{\Delta,W}^{\neg\text{SCER}}}(y) &= \left(\text{PMF}_H(W \cdot g_{\Delta,\neg\text{SCER}}^{-1}(y)) * \text{PMF}_{N_\Delta}(Wy)\right)^{*W} \\ &= \text{PMF}_H(W \cdot g_{\Delta,\neg\text{SCER}}^{-1}(y))^{*W} * \text{PMF}_{N_\Delta}(Wy)^{*W} \\ &= \text{PMF}_H(W \cdot g_{\Delta,\neg\text{SCER}}^{-1}(y))^{*W} * \text{PMF}_{N_{\Delta,W}}(y) \ ; && (5.32) \\ \text{PMF}_{Y_{\Delta,W}^{\neg\text{SCER}}}(y) &= ((\text{PMF}_H \circ W g_{\Delta,\neg\text{SCER}}^{-1})^{*W} * \text{PMF}_{N_{\Delta,W}})(y) \ . && (5.33) \end{aligned}$$

With Eq. (5.33), I can compute the actual distribution under the Non-SCER adversarial model. Because of the CLT effect, the spread of the distributions of $Y_{\Delta,W}$ distribution will shrink, enabling tight security levels even with noise. I note that the computational order should be carefully constructed for efficiency. The normal term's repeated convolution can be computed in closed form. The hypergeometric term can be computed on order $\log W$ via repeated squaring of the convolution operation. Moreover, exploiting the closed-form convolution for the normal term allows the discretization to happen at the last computation step, reducing computation and discretization precision error.

For the $Y_\Sigma^{\neg\text{SCER}}$ case, I repeat nearly the same procedure:

$$\begin{aligned}
g_{\Sigma,\neg\text{SCER}}(h|n,r,s) &= \frac{1}{2(n-r)}(2n-2r-4s+4h)\ ; && (5.34)\\
Y_\Sigma^{\neg\text{SCER}} &= k_\Sigma \cdot R_\Sigma * S^{\neg\text{SCER}} && (5.35)\\
&= k_\Sigma \cdot (R_-^w + R_-) * (\sqrt{P} R^{\neg\text{SCER}} + N)\\
&= k_\Sigma \sqrt{P} \cdot (R_-^w * R^{\neg\text{SCER}} + R_- * R^w) + k_\Sigma \cdot (R_-^w + R_-) * N\\
&= \frac{1}{2(n-r)} \cdot (2n-2r-4s+4H) + N_\Sigma\ ;\\
Y_\Sigma^{\neg\text{SCER}} &= g_{\Sigma,\neg\text{SCER}}(H) + N_\Sigma\ ; && (5.36)
\end{aligned}$$

$$\begin{aligned}
\text{PMF}_{Y_\Sigma^{\neg\text{SCER}}}(y) &= \text{PMF}_H(g_{\Sigma,\neg\text{SCER}}^{-1}(y)) * \text{PMF}_{N_\Sigma}(y)\ ; && (5.37)\\
\text{PMF}_{Y_{\Sigma,W}^{\neg\text{SCER}}}(y) &= ((\text{PMF}_H \circ W g_{\Sigma,\neg\text{SCER}}^{-1})^{*W} * \text{PMF}_{N_{\Sigma,W}})(y)\ . && (5.38)
\end{aligned}$$

### 5.3.2 Central Limit Theorem Approximation

In the prior section, I derive the distributions for the receiver authentication statistics in the presence of a Non-SCER adversary. Having the distributions on hand makes the security analysis easy to complete since the distributions enable one to specify a threshold and compute the PMD and PFA. However, because receivers will aggregate large numbers of watermarks, applying the CLT may be helpful. After applying the CLT, the distributions have closed-form solutions. For instance, the GNSS could use the CLT formulations to search for convenient parameters relevant to the GNSS and then verify using the actual distributions that the design meets the required specifications.

$Y_\Delta^{\neg\text{SCER}}$ and $Y_\Sigma^{\neg\text{SCER}}$ are clearly dependent through $H$. However, when conducting an average among $W$ vectors of $Y_\Delta$s and $Y_\Sigma$s, the $W$ vectors are independent, identically distributed, and have finite variance. Therefore, the multivariate CLT applies when operating on a large number of averages on $Y_\Delta$ and $Y_\Sigma$.

Suppose the receiver will average $W$ $Y_\Delta$s and $Y_\Sigma$s. Noting that $g_{\Delta,\neg\text{SCER}}$ and

$g_{\Sigma,\neg\mathrm{SCER}}$ are linear functions of $h$. For $Y_\Delta$:

$$\begin{aligned}
\mathbb{E}\left[Y_{\Delta,W}^{\neg\mathrm{SCER}}\right] &= \mathbb{E}\left[Y_{\Delta}^{\neg\mathrm{SCER}}\right] \\
&= \mathbb{E}\left[g_{\Delta,\neg\mathrm{SCER}}(H) + N_\Delta\right] \\
&= g_{\Delta,\neg\mathrm{SCER}}\left(\mathbb{E}\left[H\right]\right) \\
&= g_{\Delta,\neg\mathrm{SCER}}\left(\frac{sr}{n}\right) \ ; && (5.39) \\
\mathbb{E}\left[Y_{\Delta,W}^{\neg\mathrm{SCER}}\right] &= \frac{2s}{n} - 1 \ . && (5.40)
\end{aligned}$$

Also,

$$\begin{aligned}
\mathbb{V}\left[Y_{\Delta,W}^{\neg\mathrm{SCER}}\right] &= \frac{1}{W}\mathbb{V}\left[Y_{\Delta}^{\neg\mathrm{SCER}}\right] \\
&= \frac{1}{W}\mathbb{V}\left[g_{\Delta,\neg\mathrm{SCER}}(H) + N_\Delta\right] \\
&= \frac{1}{W}\mathbb{V}\left[g_{\Delta,\neg\mathrm{SCER}}(H)\right] + \frac{1}{W}\mathbb{V}\left[N_\Delta\right] \\
&= \frac{1}{W}\frac{4}{r^2}\cdot\mathbb{V}\left[H\right] + \frac{1}{W}\cdot\frac{1}{r}\cdot\frac{n}{FT}\cdot\frac{\sigma^2}{P} \\
&= \frac{1}{W}\frac{4}{r^2}\cdot\frac{sr}{n}\cdot\frac{n-r}{n}\cdot\frac{n-s}{n-1} + \frac{1}{W}\cdot\frac{1}{r}\cdot\frac{n}{FT}\cdot\frac{\sigma^2}{P} \ ; \\
\mathbb{V}\left[Y_{\Delta,W}^{\neg\mathrm{SCER}}\right] &= \frac{1}{W}\frac{4}{r}\cdot\frac{s}{n}\cdot\frac{n-r}{n}\cdot\frac{n-s}{n-1} + \frac{1}{W}\cdot\frac{1}{r}\cdot\frac{n}{FT}\cdot\frac{\sigma^2}{P} \ . && (5.41)
\end{aligned}$$

And for $Y_\Sigma$:

$$\begin{aligned}
\mathbb{E}\left[Y_{\Sigma,W}^{\neg\mathrm{SCER}}\right] &= \mathbb{E}\left[Y_{\Sigma}^{\neg\mathrm{SCER}}\right] \\
&= \mathbb{E}\left[g_{\Sigma,\neg\mathrm{SCER}}(H) + N_\Sigma\right] \\
&= g_{\Sigma,\neg\mathrm{SCER}}\left(\mathbb{E}\left[H\right]\right) \\
&= g_{\Sigma,\neg\mathrm{SCER}}\left(\frac{sr}{n}\right) \ ; && (5.42) \\
\mathbb{E}\left[Y_{\Sigma,W}^{\neg\mathrm{SCER}}\right] &= 1 - \frac{2s}{n-r} + \frac{2sr}{(n-r)n} \ ; && (5.43)
\end{aligned}$$

$$\begin{aligned}
\mathbb{V}\left[Y_{\Sigma,W}^{\neg\text{SCER}}\right] &= \frac{1}{W}\mathbb{V}\left[Y_{\Sigma}^{\neg\text{SCER}}\right] \\
&= \frac{1}{W}\mathbb{V}\left[g_{\Sigma,\neg\text{SCER}}(H) + N_{\Sigma}\right] \\
&= \frac{1}{W}\mathbb{V}\left[g_{\Sigma,\neg\text{SCER}}(H)\right] + \frac{1}{W}\mathbb{V}\left[N_{\Sigma}\right] \\
&= \frac{1}{W}\frac{4}{(n-r)^2}\cdot\mathbb{V}\left[H\right] + \frac{1}{W}\cdot\frac{1}{n-r}\cdot\frac{n}{FT}\cdot\frac{\sigma^2}{P} \\
&= \frac{1}{W}\frac{4}{(n-r)^2}\cdot\frac{sr}{n}\cdot\frac{n-r}{n}\cdot\frac{n-s}{n-1} + \frac{1}{W}\cdot\frac{1}{n-r}\cdot\frac{n}{FT}\cdot\frac{\sigma^2}{P}\ ; \\
\mathbb{V}\left[Y_{\Sigma,W}^{\neg\text{SCER}}\right] &= \frac{1}{W}\frac{4}{n-r}\cdot\frac{sr}{n}\cdot\frac{1}{n}\cdot\frac{n-s}{n-1} + \frac{1}{W}\cdot\frac{1}{n-r}\cdot\frac{n}{FT}\cdot\frac{\sigma^2}{P}\ . \qquad (5.44)
\end{aligned}$$

By computing a derivative of $\mathbb{V}\left[Y_{\Delta,W}^{\neg\text{SCER}}\right]$ and $\mathbb{V}\left[Y_{\Sigma,W}^{\neg\text{SCER}}\right]$, one can verify that the maximum variances occur when $s = \frac{n}{2}$.

For the covariance:

$$\begin{aligned}
&\operatorname{Cov}\left[Y_{\Delta,W}^{\neg\text{SCER}}, Y_{\Sigma,W}^{\neg\text{SCER}}\right] = \frac{1}{W}\operatorname{Cov}\left[Y_{\Delta}^{\neg\text{SCER}}, Y_{\Sigma}^{\neg\text{SCER}}\right] \\
&= \frac{1}{W}\mathbb{E}\left[Y_{\Delta}^{\neg\text{SCER}}\cdot Y_{\Sigma}^{\neg\text{SCER}}\right] - \frac{1}{W}\mathbb{E}\left[Y_{\Delta}^{\neg\text{SCER}}\right]\mathbb{E}\left[Y_{\Sigma}^{\neg\text{SCER}}\right] \\
&= \frac{1}{W}\mathbb{E}\left[Y_{\Delta}^{\neg\text{SCER}}\cdot Y_{\Sigma}^{\neg\text{SCER}}\right] - \frac{1}{W}\left(\frac{2s}{n}-1\right)\left(1-\frac{2s}{n-r}+\frac{2sr}{(n-r)n}\right)\ .
\end{aligned}$$

Both $Y_{\Delta,W}^{\neg\text{SCER}}$ and $Y_{\Sigma,W}^{\neg\text{SCER}}$ are the sum of (1) affine functions of the same hypergeometric distribution and (2) independent normal distributions. Each normal distribution is independent of the other normal distribution and that hypergeometric distribution.

Therefore, only one term from the product expectation remains:

$$
\begin{aligned}
&= \frac{1}{W}\mathbb{E}\left[g_{\Delta,\neg\mathrm{SCER}}(H)\cdot g_{\Sigma,\neg\mathrm{SCER}}(H)\right]- \\
&\qquad \frac{1}{W}\left(\frac{2s}{n}-1\right)\left(1-\frac{2s}{n-r}+\frac{2sr}{(n-r)n}\right)\ ; \\
&\mathrm{Cov}\left[Y_{\Delta,W}^{\neg\mathrm{SCER}}, Y_{\Sigma,W}^{\neg\mathrm{SCER}}\right] = \\
&\quad \frac{1}{W}\sum_{h=0}^{\min(r,s)} g_{\Delta,\neg\mathrm{SCER}}(h)\cdot g_{\Sigma,\neg\mathrm{SCER}}(h)\cdot \mathrm{PMF}_H(h)- \\
&\qquad \frac{1}{W}\left(\frac{2s}{n}-1\right)\left(1-\frac{2s}{n-r}+\frac{2sr}{(n-r)n}\right)
\end{aligned}
\tag{5.45}
$$

From my experience, the computed $\mathrm{Cov}\left[Y_{\Delta,W}^{\neg\mathrm{SCER}}, Y_{\Sigma,W}^{\neg\mathrm{SCER}}\right]$ is so small that the ellipse rotation effect in figures like Fig. 5.7 is imperceptible to the human eye. As discussed in Section 5.5, since the CLT will be used as approximations for an initial search on the parameters and verified later, this covariance is not a significant concern.

## 5.3.3 Experimental Validation

In this section, I provide experimental evidence that the distributions from Section 5.3.1 are correct. First, I discuss a Monte Carlo simulation. Second, I discuss experimentation with a software-defined radio (SDR) with GPS C/A data.

### Monte Carlo Validation

To simulate the statistical distributions via Monte Carlo methods, I utilize the signal model from Eq. (5.21). For 10000 trials, I simulated 10 watermarked GPS PRN1 ranging codes. Each ranging code included a random watermark and simulated AWGN noise. Furthermore, each ranging code included $FT$ measurements. To simulate the adversary, the adversary generated its own watermarked ranging code.

Both the authentic and the spoofed watermark signals were fed into the $Y_\Delta$ and $Y_\Sigma$ filters. For each trial, the 10 $Y_\Delta$ and $Y_\Sigma$ results were averaged together to produce the data in the histogram in Fig. 5.5. The predicted authentic distributions come from

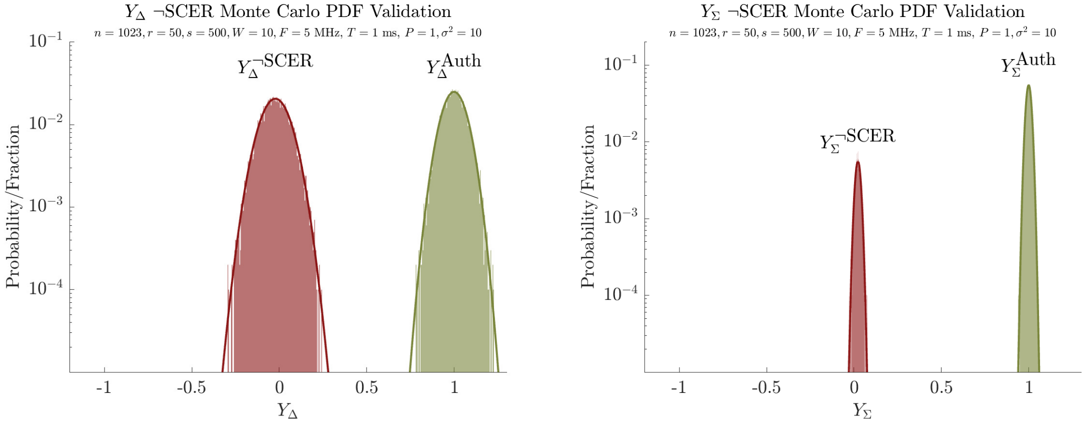


Figure 5.5: Non-SCER Statistic Distribution Monte Carlo Validation.

The figure depicts the results of a Monte Carlo experiment to validate the distributions from Eqs. (5.33) and (5.38) computed via repeated convolution. The parameters were selected to show the efficacy of the distribution predictions and are $n = 1023, r = 50, s = 500, P = 1, \sigma^2 = 10, F = 5$ MHz, $T = 1$ ms, $W = 10$. The experiment included 10000 trials with each trial containing $W = 10$ watermarked PRN1 ranging code with simulated noise.

Section 5.2.2. The distributions of predicted Non-SCER were computed via repeated convolution with Eqs. (5.33) and (5.38).

The parameters selected for simulation in Fig. 5.5 are not ideal for a real watermarking scheme. Rather, they were selected judiciously to demonstrate the efficacy of the computed distributions. For a more realistic scheme, such as that from Section 5.5 where $W = 6000$, the distributions would severely concentrate on the means, rendering such a figure uninteresting.

### GPS C/A SDR Validation

I found an opportunity to demonstrate the distributions' efficacy with actual data by modifying an existing SDR [23]. I collected GPS C/A data with a Universal Software Radio Peripheral providing raw 16-bit I and Q samples at 25 Mhz. Fig. 5.6 provides the results of my experiment.

At the time of writing, no watermarked signals were broadcast, so I tested utilizing a "mirrored" situation. With a real watermarking signal authentication scheme, the

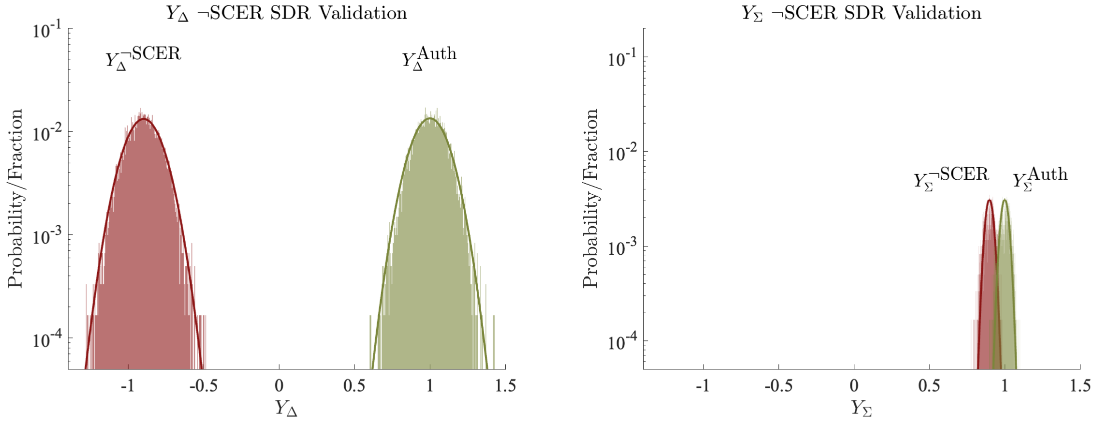


Figure 5.6: Non-SCER Distribution Validation With Experimental SDR.

In this experiment, I compare real-world measurements of $Y_\Delta$ and $Y_\Sigma$ with the predicted distributions from Section 5.3.1 computed via repeated convolution. The experiment was conducted with post-processed GPS C/A observations made with a 25 MHz SDR based on the mirrored situation described in Section 5.3.3. Over a 60-second time period, I measured 6000 ranging codes, each computed over a 10 ms integration within a converged tracking loop. Each millisecond contains its own mirrored watermark, so each $Y$ is related to 10 independent watermarks. In this scenario, the adversary has elected to randomly guess $s = 52$ chips to invert per millisecond against $r = 52$ chips inverted by the provider.

receiver tracks a watermarked signal using the original ranging code. Instead, to construct the mirrored situation, I programmed the SDR to track the real signal with a watermarked replica. Provided $r = s$, the net result is the negative $Y$s. Because this experiment requires the $r = s$ election, the two distributions (authentic and ¬SCER) of $Y_{\Delta,W}$ are far apart, and the two distributions of $Y_{\Sigma,W}$ are close together. The $P$ and $\sigma^2$ were estimated via the methods discussed in Section 5.2.3.

### 5.3.4 Expectation Trajectory and Decision Theory

The receiver must make a decision: authentic or not authentic. In Decision Theory, typically, one would test a single statistic against a threshold. Whether that threshold is met determines the decision, and one hopes such a test has specificity and sensitivity. Or, in the security context, one hopes the test has a low spoofing PMD and a low authentic PFA.

For watermarking, there are the two statistics from Section 5.2.1. From Section 5.2.2, one can compute the statistics' distribution in the authentic case in closed form. From Section 5.3.1, one can compute the distributions under a Non-SCER adversary via convolution. And from Section 5.3.2, one can compute CLT approximations of Section 5.3.1 to help the initial design, after which the design can be verified with Section 5.3.1. From these distributions, one can construct a simple hypothesis test and determine the PMD and PFA.

The advantage of using these two statistics is apparent in Section 5.4.6, where the adversarial $Y_\Delta^{\text{HDSCER}}$ and $Y_\Sigma^{\text{HDSCER}}$ are independent. For the Non-SCER, the two statistics are conditionally independent on $h$. This means that when evaluating the missed-detection and false-alarm hypotheses via integration, the primary integration axis should be $h$ (and then the secondary axis should be $y$). For instance, with decision thresholds $\alpha_\Delta$ and $\alpha_\Sigma$, the PMDs for each statistic used separately would be

$$\Pr\left(Y_\Delta^{\neg\text{SCER}} > \alpha_\Delta\right) = \sum_{h=0}^{r} \Pr(Y_\Delta^{\neg\text{SCER}} > \alpha_\Delta \mid h) \Pr(h) \; ; \tag{5.46}$$

$$\Pr\left(Y_\Sigma^{\neg\text{SCER}} > \alpha_\Sigma\right) = \sum_{h=0}^{r} \Pr(Y_\Sigma^{\neg\text{SCER}} > \alpha_\Sigma \mid h) \Pr(h) \; . \tag{5.47}$$

After conditioning on $h$, each remaining probability term results from a simple normal distribution and the threshold.

For the $Y_\Delta$ and $Y_\Sigma$ statistics, Fig. 5.7 plots the authentic and Non-SCER hypothesis CLT distributions for a simple case to convey trends for design intuition. Because the Non-SCER adversary may elect $s$, the number of spoofed inversions, one must account for each $s$. Fig. 5.7 plots the $Y_\Delta$ and $Y_\Sigma$ Non-SCER distribution trajectory over the adversary's $s$ election. The trajectory derives from the means derived from Section 5.3.2, and the figure depicts the 1-sigma variance ellipse with a single watermark $W = 1$. This trajectory reveals why statistics $Y_\Delta$ and $Y_\Sigma$ must be used simultaneously: if only one is used, the adversary can elect an $s$ to spoof the single $Y$ utilized.

Fig. 5.7 follows the scheme parameter choices from Section 5.5: $n = 1023$ and

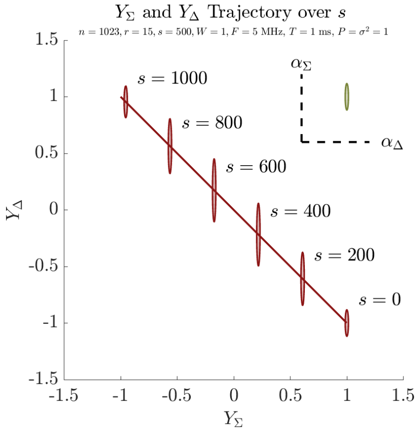


Figure 5.7: Non-SCER Statistic Trajectory Over Adversarial Strategy.

The Non-SCER adversary may elect any $s$ strategy when attempting to break authentication security with a watermark. The figure depicts how the adversary's $s$ election effects the resulting $Y_\Delta$ and $Y_\Sigma$ using the CLT simplification from Section 5.3.2. The figure shows the 1-sigma ellipse for unrealistic noise conditions and with a single watermark to demonstrate the trajectory itself and how the variances change over the trajectory. In a real scheme where the receiver averages over 1000s of watermarks (e.g., Section 5.5), the CLT effect shrinks the distributions into dots (making this figure less interesting). The authentic distribution in **green** centers at $(1, 1)$, whereas the non-SCER in **cardinal** traces a straight-line trajectory over $s$.

$r = 15$. For a real receiver aggregating 1000s of watermarks (e.g., $W = 6000$ in Section 5.5), the CLT effect will shrink the spread of the distributions into dots, making Fig. 5.7 less interesting. To demonstrate the trends, Fig. 5.7 assumes SNR $\frac{P}{\sigma^2} = 1$ and that the receiver observes over a single $W = 1$ watermark. And Fig. 5.7 assumes a $F = 5$ MHz radio with a 1 ms integration time. These watermark scheme parameters are unrealistic to show the variance trend over the $s$ trajectory.

The adversaries' choice of $s$ will be based on the receiver's decision line. The first approach to determine the PMD and PFA could be the following. First, the receiver elects a decision line. Second, the receiver performs a search over $s$ to find the $s$ with the best PMD. Assuming the receiver elects a decision line like that in Section 5.3.4, this would mean $s \approx 500$. Over the search on $s$, the receiver will compute the PMD leveraging that $Y_\Delta^{\neg\text{SCER}}$ and $Y_\Sigma^{\neg\text{SCER}}$ are conditionally independent on $h$. This results with how knowledge of $h$ captures all the dependent information between $Y_\Delta^{\neg\text{SCER}}$ and

$Y_{\Sigma}^{\neg\mathrm{SCER}}$ since the remaining normal components are AWGN. Therefore, the receiver will determine whether the PMD for the adversary's best $s$ is acceptable via

$$\mathrm{PMD}_{Y_{\Delta}>\alpha_{\Delta},Y_{\Sigma}>\alpha_{\Sigma}}^{\neg\mathrm{SCER}} = \sum_{h=0}^{r} \Pr(Y_{\Delta}^{\neg\mathrm{SCER}} > \alpha_{\Delta} \mid h) \Pr(Y_{\Sigma}^{\neg\mathrm{SCER}} > \alpha_{\Sigma} \mid h) \, \mathrm{PMF}_{H}(h) \; . \tag{5.48}$$

Fig. 5.7 depicts when $\alpha_{\Delta}$ and $\alpha_{\Delta}$ are constant, causing the decision line to separate an authentic corner about $(1, 1)$, allowing Eq. (5.48) to be a simple product. However, the decision line could be curved (e.g., a circle around $(1, 1)$) for a potentially more specific and sensitive hypothesis test. Section 5.5 elects the decision line $Y_{\Delta} + Y_{\Sigma} = 1$. And to generalize for the $W$ case, the integration should be over $H^{*W}$ rather than $H$. Because the terms after conditioning are integrations over uncorrelated normal variables, the PMD can be computed as a sum of CDF calls:

$$\begin{aligned}
\mathrm{PMD}_{W,Y_{\Delta}+Y_{\Sigma}\geq 1}^{\neg\mathrm{SCER}} &= \sum_{h=0}^{r\cdot W} \Pr(Y_{\Delta,W}^{\neg\mathrm{SCER}} + Y_{\Sigma,W}^{\neg\mathrm{SCER}} \geq 1 \mid H^{*W} = i) \Pr(H^{*W} = i) \qquad (5.49)\\
&= \sum_{h=0}^{r\cdot W} \Pr(g_{\Delta,\neg\mathrm{SCER}}(i) + N_{\Delta} + g_{\Sigma,\neg\mathrm{SCER}}(i) + N_{\Sigma} \geq 1) \Pr(H^{*W} = i)\\
&= \sum_{h=0}^{r\cdot W} \Pr(N_{\Delta} + N_{\Sigma} \geq 1 - g_{\Delta,\neg\mathrm{SCER}}(i) - g_{\Sigma,\neg\mathrm{SCER}}(i)) \Pr(H^{*W} = i) \; ;\\
\mathrm{PMD}_{W,Y_{\Delta}+Y_{\Sigma}\geq 1}^{\neg\mathrm{SCER}} &= \sum_{h=0}^{r\cdot W} \mathrm{CDF}_{N(0,\mathbb{V}[N_{\Delta}]+\mathbb{V}[N_{\Sigma}])}^{\text{Upper Compliment}}(1 - g_{\Delta}(i) - g_{\Sigma}(i)) \Pr(H^{*W} = i) \; . \qquad (5.50)
\end{aligned}$$

And for the PFA, simply

$$\begin{aligned}
\mathrm{PFA}_{W,Y_{\Delta}+Y_{\Sigma}\geq 1} &= \Pr(Y_{\Delta}^{\mathrm{auth}} + Y_{\Sigma}^{\mathrm{auth}} < 1) \; ;\\
\mathrm{PFA}_{W,Y_{\Delta}+Y_{\Sigma}\geq 1} &= \mathrm{CDF}_{\mathcal{N}(2,\mathbb{V}[N_{\Delta}]+\mathbb{V}[N_{\Sigma}])}(1) \; . \qquad (5.51)
\end{aligned}$$

As a second approach, the receiver could select the thresholds for $Y_{\Delta}$ and $Y_{\Sigma}$ separately, which was the approach for [5, 6]. $Y_{\Sigma}$ relates to the receiver's ability to track the signal, and a $Y_{\Sigma} \to 0$ is when the adversary provides a completely random ranging code. At this point, the receiver should not be able to track the signal. Therefore, the receiver could specify a threshold on $Y_\Sigma$ related to the minimum needed for tracking. And then, the receiver can elect a threshold on $Y_\Delta$ that meets missed-detection requirements by directly finding the needed boundary on Eq. (5.31) with just Eq. (5.46).

If the computed missed detection probability is not acceptable, the receiver can examine a larger amount of watermarks together (i.e., increase $W$, possibly negatively affecting the TTA), increase its sampling frequency $F$, or modify its decision line, possibly increasing the probability of false alarm. Or the GNSS designer could increase $r$. Additional discussion on designing the scheme parameters is in Section 5.5.

## 5.4 SCER Adversary Security

With the SCER adversarial model, the adversary may observe the authentic watermarked signal, apply signal processing to attempt to deduce the watermark, and then transmit a spoofed signal with its best-estimated watermark as a forgery. Fig. 5.8 provides a conceptual diagram of an SCER attack for the combinatorial-watermarking context. Among GNSS spoofing adversaries, SCER-capable adversaries are considerably more sophisticated and complicated [18]. In some contexts, GNSS authentication schemes that prohibit all but SCER-capable adversaries are sufficient spoofing deterrents to meeting anti-spoofing requirements. Or the analysis of this section enables authorities to predict the required radio equipment to break the scheme, for instance, the size of a directed antennae needed for which authorities can monitor near an airport. However, even when cryptography is incorporated into GNSS signals, GNSS signals will still remain vulnerable to SCER attacks.

SCER attacks are difficult for multiple reasons. One reason is that the GNSS signal is below the thermal noise floor, and estimating security chips requires sophisticated (and likely arduous) radio equipment (e.g., high-gain antennae). A second reason is that the adversary must transport the estimate of the security chips to a transmitting antenna within a sufficiently short time to avoid detection by the receiver's GIC, as discussed in Section 2.1.2. A third reason is that the cryptographic construction

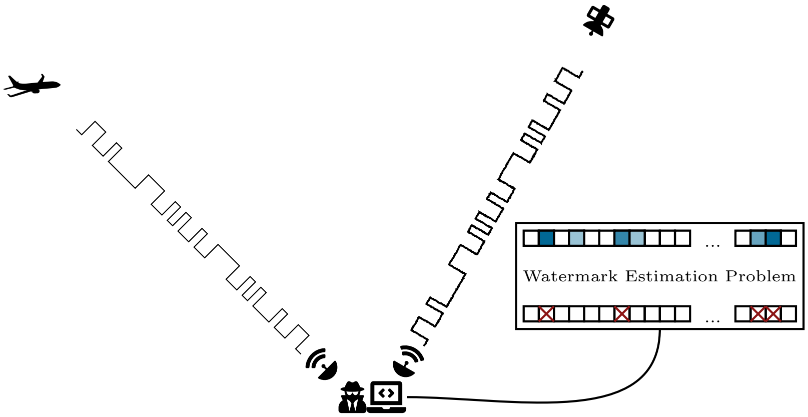


Figure 5.8: Conceptual Diagram Of The Components Of An SCER Attack.

The adversary attempts to observe the watermark directly in the signal and then replay a watermarked signal to spoof a receiver. Within the adversary's signal processing, the diagram portrays the adversary attempting to use its measurements of the true signal to construct a single watermark likely to spoof without detection by the receiver. The top row of boxes represents a collection of measured inverted-chip likelihoods among a single watermarked ranging code. The varying hues of **blue** represent the soft information provided by the likelihood (i.e., the darker the **blue**, the higher the inverted likelihood). The bottom row of boxes represents the adversary's decision to watermark. The adversary can elect to invert any number of chips based on its measurements.

limits the effectiveness of advanced decision algorithms beyond exhaustive search among an enormous search space. Like with [57], I will assume that the adversary has access to advanced radio equipment and has zero latency: no delay among its observation antenna, the watermark decision-making algorithm (though a practical computer must be capable of computing the decision), and the replaying antenna.

To evaluate the security of a watermarking scheme against an SCER adversary, first, one must understand chip estimation theory, which is discussed in Section 5.4.1. Section 5.4.2 discusses the theory of aggregating individual chip estimations. Section 5.4.3 discusses the definition of an SCER adversary with a limiting assumption. This assumption is relaxed in Section 5.4.7. However, the mathematics from Section 5.4.3 will provide a foundation for GNSS design. Section 5.4.4 provides the CLT simplification of Section 5.4.3. Section 5.4.5 discusses Monte Carlo experiments to validate the derived distributions of Section 5.4.3. And Section 5.4.6 discusses the authentication decision theory similar to Section 5.3.4.

### 5.4.1 Chip Estimation Theory

[57] presents a detection statistic for hypotheses regarding the authenticity of a single chip under various SCER-capable adversaries, which this section abbreviates and customizes for this application. While the adversary will need to make a decision when considering all $n$ chips in a watermark, this section considers the situation when the adversary makes chip decisions individually, assuming independence among the chips. After measuring the chip (potentially with multiple samples per chip), the adversary makes a decision, whether the chip is not inverted or inverted, based on a chip estimation model and a threshold.

Under the standard $\sigma^2$-noise-power AWGN assumption for a binary phase shift key (BPSK) signal, the signal constellation diagram points are separated by $2\sqrt{P}$, distributed normally with standard deviation $\sigma$, and 0 is halfway between them, as in Fig. 5.9. Without loss of generality, suppose that a noninverted chip will center at $\sqrt{P}$ and an inverted chip will center at $-\sqrt{P}$ in the baseband measurement. This is practically achieved by wiping off the ranging code element-wise and multiplying the signal $S$ by the replica $R$.

The chip-estimating adversary must elect a decision boundary $\alpha$. Halfway would be a good choice, assuming a uniform prior between inverted and noninverted. However, that will never be the case for a watermarked signal because the number of watermarked chips must be less than the number of non-watermarked chips so that receivers can track the signal. So, the adversary could elect to incorporate the prior probability that a chip is inverted (e.g., a Maximum Likelihood or Maximum A Posteriori Decision). The adversary could elect ***any*** decision boundary, so one must parameterize the adversary election just like $s$ in Section 5.3.

For the moment, let us assume that the adversary makes one measurement per chip. Suppose the adversary elects the decision $\alpha$. Given a boundary $\alpha$, the probabilities of error $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$ given whether the chip is inverted or not inverted are

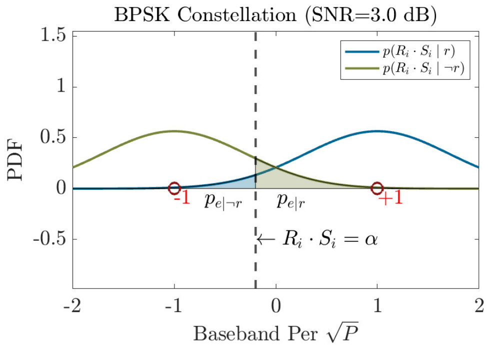


Figure 5.9: Conceptual Figure Of BPSK Chip Estimation Model.

The figure depicts the BPSK model probability density function (PDF) for estimation of a single chip after element-wise multiplying by the unwatermarked replica. For the non-watermarked chip hypothesis, the constellation point will be 1. For the watermarked chip hypothesis, the constellation point will be -1. The diagram includes the probability density functions with an SNR of 3dB to provide an intuitive depiction. The adversary may elect any decision boundary $\alpha$ (e.g., a Maximum Likelihood or Maximum A Posterior Decision). The probabilities of errors are labeled, given the decision boundary and noise model.

provided in Eqs. (5.52) and (5.53):

$$p_{\mathrm{e}|r} = \int_{\alpha}^{\infty} \mathrm{PDF}_{\mathcal{N}(-\sqrt{P},\sigma^2)}(x)dx \tag{5.52}$$

$$p_{\mathrm{e}|\neg r} = \int_{-\infty}^{\alpha} \mathrm{PDF}_{\mathcal{N}(\sqrt{P},\sigma^2)}(x)dx \ . \tag{5.53}$$

By parameterizing the chip estimation by $\alpha$, I account for any prior-chip-inversion probability strategy by the adversary.

To account for multiple measurements, the adversary could average the individual measurements to form a better measurement. In that case, the variance decreases by a factor of the number of measurements per chip. This has the same effect, increasing the incident SNR for the single measurement per chip case. Without loss of generality and for mathematical brevity, I assume and formulate with a single measurement per chip hereafter.

With Eqs. (5.52) and (5.53), the PDF model is abstracted into $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$. In the next sessions, the model will be a function of $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$, meaning the derivations will apply even without the BPSK model. For instance, the security tester could measure $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$ directly rather than estimating the adversary's SNR and then modeling $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$. But for formulating $\alpha$ trajectories of $\mathbb{E}\left[Y_\Delta\right]$ and $\mathbb{E}\left[Y_\Sigma\right]$ just like the $s$ trajectories of Fig. 5.7, I assume the BPSK model with adversary decision election $\alpha$ (see Section 5.4.6).

### 5.4.2 Adversarial Watermark Estimation Theory

As conceptually diagrammed in Fig. 5.8, the adversary must make a decision based on its chip measurements. While the adversary has access to measurements of all the $n$ chips that compose a watermarked ranging code, it is unclear how the adversary could exploit the $n$ chip measurements together. Suppose the adversary forms a Bayesian watermark detector of all of the chip measurements under each hypothesis $R^i$, where $i$ indexes each of the $\binom{n}{r}$ possible watermarks. In the context of deriving an optimal detector, there is no reason to penalize the detection of one watermark over another among the $R^i$. Moreover, because each $R^i$ is chosen uniformly, the prior probability $\Pr(R^i)$ is also uniform. Therefore, the optimal detector will be the maximum likelihood detector:

$$\begin{aligned}\Pr(R^i \mid S) &= \frac{\Pr(S \mid R^i)\Pr(R^i)}{\Pr(S)} \\ \Pr(R^i \mid S) &\propto \Pr(S \mid R^i) \qquad (5.54)\end{aligned}$$

and the optimal decision rule is

$$\arg\max \; \Pr(S \mid R^i) \; . \qquad (5.55)$$

From the BPSK model from Section 5.4.1, $\Pr(S \mid R^i)$ would be a multivariate normal distribution with a closed-form expression. Given $R^i$, the remaining randomness is just the AWGN variables, so $\Pr(S \mid R^i)$ is an uncorrelated multivariate normal distribution with varying 1s and $-1$s as variable means. However, fortunately for authentication security, the $\binom{n}{r}$ number of possible watermarks is enormous, rendering such a watermark detector infeasible. For instance, in Section 5.5, the watermark proposed has $\binom{1023}{15} > 2^{109}$ potential watermarks per millisecond. However, error correction code theory provides insight into how detectors could help the SCER adversary.

A Combinatorial Watermark can be thought of as an error correction code with $\binom{n}{r}$ valid words among $2^n$ possible bit transmissions. In a normal error correction code, there will be patterns among multiple words that a detector can leverage to help narrow down the $2^n$ to the correct word among the $\binom{n}{r}$. But with the cryptographic security of the Combinatorial Watermark, knowledge of such patterns would break the underlying cryptographic primitives. However, the error correction code theory provides a useful information framework for moving forward. An error correction decoder can utilize hard or soft information for hard or soft decisions, respectively.

When a hard decision error correction decoder makes a decision on a bit, it reports a binary decision. Either the bit is 1 or 0. When a soft decision error correction decoder makes a decision on a bit, it reports additional soft information. For instance, it could report probabilities for whether the bit is 1 or 0. Reporting 50/50 would indicate no confidence; reporting 1/0 would indicate perfect confidence. The soft decision error correction decoder can make a holistic determination based on the soft information from all the bits in its decision.

When examining an hard-decision security code estimation and replay (HDSCER) adversary, similar mathematical distributions to Section 5.3.1 can be derived. An HDSCER adversary makes a hard decision about each of the chips within the watermark ranging code without regard to the $r$ structure of the watermark. Rather than the hypergeometric distributions from Section 5.3.1, the HDSCER adversary relates to binomial distributions, providing a mathematical pathway for analysis. Section 5.4.3 discusses the HDSCER adversary. While it is more difficult to apply soft-decision theory to error correction codes, it is even harder in this context where the watermarking code is cryptographically unhelpful. However, Section 5.4.7 presents a soft-decision SCER adversary that provides a small advantage to the HDSCER adversary.

### 5.4.3 Hard Decision SCER Statistical Distributions

The HDSCER utilizes the chip estimation theory from Section 5.4.1 to make chip-independent decisions about whether a particular chip is inverted. The name "hard-decision" is inspired by error correction code theory (see Section 5.4.2). Because the HDSCER adversary makes chip-independent decisions, it completely ignores the $r$ structure of the watermark. If half of the chips are believed to be inverted, the adversary's spoof has those chips inverted, regardless of the specified $r$.

From Section 5.4.1, the adversary may elect a decision boundary $\alpha$ on its chip estimation decision problem. From $\alpha$, suppose the probabilities of chip estimation error are $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$. After the distribution of the hash point, the individual chip measurements become conditionally independent because the hash point captures all of the information of which $r$ is inverted. With this independence, modeling the $r$ inverted chips and the $(n-r)$ noninverted chips can be done independently after hash point distribution, which is when the receiver will do signal processing. The probability that the adversary will guess the $r$ correctly and the $n-r$ correctly follows separate binomial distributions.

Let $B_r$ be the number of inverted chips among $r$ that the adversary observes correctly, which follows a binomial distribution. Let $B_{\neg r}$ be the number of noninverted chips among $n-r$ that the adversary observes incorrectly, which also follows a binomial distribution. In both cases, $B$ is a number of chips the adversary elects to invert. Like Section 5.3.1, let $R^{\mathrm{HDSCER}}$ be the adversary-selected replica in its spoof. First, let's derive some useful convolutions:

$$B_r \sim \mathcal{B}(r, 1-p_{\mathrm{e}|r}) \; ; \tag{5.56}$$

$$B_{\neg r} \sim \mathcal{B}(n-r, p_{\mathrm{e}|\neg r}) \; ; \tag{5.57}$$

$$\mathrm{PMF}_{\mathcal{B}(n,p)}(b) = \binom{n}{b} p^b (1-p)^{n-b} \; ; \tag{5.58}$$

$$R_-^w * R^{\mathrm{HDSCER}} = (n - 2r + 2B_r - 2B_{\neg r})\frac{FT}{n} \; ; \tag{5.59}$$

$$R_- * R^{\mathrm{HDSCER}} = (n - 2B_r - 2B_{\neg r})\frac{FT}{n} \; . \tag{5.60}$$

Consider the case where the adversary makes no inversions and $FT = n$: $R^w_- * R^{\text{HDSCER}} = R^w_- * R = n - 2r$. Each time the adversary correctly estimates an inverted chip, $R^w_- * R^{\text{HDSCER}}$ increases by 2; each time the adversary incorrectly estimates a noninverted chip, $R^w_- * R^{\text{HDSCER}}$ decreases by 2, arriving at Eq. (5.59). Similarly, consider the case where the adversary makes no inversions and $FT = n$: $R_- * R^{\text{HDSCER}} = R_- * R = n$. Each time the adversary correctly estimates an inverted chip, $R_- * R^{\text{HDSCER}}$ decreases by 2; each time the adversary incorrectly estimates a noninverted chip, $R_- * R^{\text{HDSCER}}$ decreases by 2, arriving at Eq. (5.60).

Now, the choice of filters $Y_\Delta$ and $Y_\Sigma$ separate the binomial distributions. From Eqs. (5.59) and (5.60) the following are useful convolutions for deriving the distributions of $Y_\Delta$ and $Y_\Sigma$ in the HDSCER case:

$$(R^w_- - R_-) * R^{\text{HDSCER}} = (-2r + 4B_r)\frac{FT}{n} \ ; \tag{5.61}$$

$$(R^w_- + R_-) * R^{\text{HDSCER}} = (2(n-r) - 4B_{\neg r})\frac{FT}{n} \ . \tag{5.62}$$

Like in Section 5.3.1, for mathematical conciseness, I will introduce

$$y = g_{\Delta,\text{HDSCER}}(b|r) = \frac{1}{2r} \cdot (4b - 2r) \ ; \tag{5.63}$$

$$y = g_{\Sigma,\text{HDSCER}}(b|r) = \frac{1}{2(n-r)} \cdot (2(n-r) - 4b) \ . \tag{5.64}$$

Feeding signal $S$ into $Y_\Delta$ yields

$$\begin{aligned}
Y_\Delta^{\text{HDSCER}} &= k_\Delta(R^w_- - R_-) * (\sqrt{P}R^{\text{HDSCER}} + N) \\
&= k_\Delta\sqrt{P}(R^w_- - R_-) * R^{\text{HDSCER}} + k_\Delta \cdot (R^w_- - R_-) * N \\
&= \frac{1}{2r}(-2r + 4B_r) + k_\Delta \cdot (R^w_- - R_-) * N \\
Y_\Delta^{\text{HDSCER}} &= g_{\Delta,\text{HDSCER}}(B_r) + N_\Delta \ ; && (5.65) \\
\text{PMF}_{Y_\Delta^{\text{HDSCER}}}(y) &= \text{PMF}_{B_r}(g^{-1}_{\Delta,\text{HDSCER}}(y)) * \text{PMF}_{N_\Delta}(y) \ ; \\
\text{PMF}_{Y_\Delta^{\text{HDSCER}}}(y) &= ((\text{PMF}_{B_r} \circ g^{-1}_{\Delta,\text{HDSCER}}) * \text{PMF}_{N_\Delta})(y) \ . && (5.66)
\end{aligned}$$

Feeding signal $S$ into $Y_\Sigma$ yields

$$\begin{aligned}
Y_\Sigma^{\mathrm{HDSCER}} &= k_\Sigma(R_-^w + R_-) * (\sqrt{P}R^{\mathrm{HDSCER}} + N) \\
&= k_\Sigma\sqrt{P}(R_-^w + R_-) * R^{\mathrm{HDSCER}} + k_\Sigma \cdot (R_-^w + R_-) * N \\
&= \frac{1}{2(n-r)}(2(n-r) - 4B_{\neg r}) + k_\Sigma \cdot (R_-^w + R_-) * N
\end{aligned}$$

$$Y_\Sigma^{\mathrm{HDSCER}} = g_{\Sigma,\mathrm{HDSCER}}(B_{\neg r}) + N_\Sigma \; ; \tag{5.67}$$

$$\mathrm{PMF}_{Y_\Sigma^{\mathrm{HDSCER}}}(y) = \mathrm{PMF}_{B_{\neg r}}(g_{\Sigma,\mathrm{HDSCER}}^{-1}(y)) * \mathrm{PMF}_{N_\Sigma}(y) \; ;$$

$$\mathrm{PMF}_{Y_\Sigma^{\mathrm{HDSCER}}}(y) = ((\mathrm{PMF}_{B_{\neg r}} \circ g_{\Sigma,\mathrm{HDSCER}}^{-1}) * \mathrm{PMF}_{N_\Sigma})(y) \; . \tag{5.68}$$

Repeating from Section 5.3.1, for the case where the receiver averages over $W$ watermarks:

$$\mathrm{PMF}_{Y_{\Delta,W}^{\mathrm{HDSCER}}}(y) = ((\mathrm{PMF}_{B_r} \circ W g_{\Delta,\mathrm{HDSCER}}^{-1})^{*W} * \mathrm{PMF}_{N_{\Delta,W}})(y) \; ; \tag{5.69}$$

$$\mathrm{PMF}_{Y_{\Sigma,W}^{\mathrm{HDSCER}}}(y) = ((\mathrm{PMF}_{B_{\neg r}} \circ W g_{\Sigma,\mathrm{HDSCER}}^{-1})^{*W} * \mathrm{PMF}_{N_{\Sigma,W}})(y) \; . \tag{5.70}$$

The $Y_\Delta$ statistic measures how well the adversary can predict where the chip inversions exist. The $Y_\Sigma$ statistic measures how good the adversary's false-inverted-chip-estimation rate is. Because the receiver will initially track with $R$, the $Y_\Sigma$ statistic also measures how well the receiver will track the spoofed signal.

As with Section 5.3.1, one can compute the actual distributions on the HDSCER adversarial model. Because of the CLT effect, the spread of the distributions of $Y$ will shrink about their expectations, which Section 5.4.4 will show are functions of $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$.

Because of the construction of $Y_\Delta$ and $Y_\Sigma$, they derive from independent binomial distributions. Therefore, these statistics are independent, which is why this chapter utilizes them. Another set of statistics would not be independent, complicating analysis.

### 5.4.4 Central Limit Theorem Approximation

Just like Section 5.3.2, applying CLT simplifications can aid the design of the scheme. For instance, the GNSS could use the CLT to search for convenient parameters relevant to the GNSS. In Section 5.4.6, I will use the CLT simplifications to intuitively show the adversary's game, and the receiver's defense, with the trends of the expectations of $Y_{\Delta}^{\text{HDSCER}}$ and $Y_{\Sigma}^{\text{HDSCER}}$.

When conducting an average among $W$ vectors of $Y_{\Delta}$s and $Y_{\Sigma}$s, the $W$ vectors are independent, identically distributed, and have finite variance. Therefore, the multivariate CLT applies when operating on averages of large numbers of $Y_{\Delta}$ and $Y_{\Sigma}$. Because all terms within $Y_{\Delta}^{\text{HDSCER}}$ and $Y_{\Sigma}^{\text{HDSCER}}$ are independent, there is no covariance for the HDSCER case (unlike in Section 5.3.2).

I will now derive the expectations and variances of the two output filter statistics under the HDSCER model. I note simply setting $p_{\text{e}|r} = p_{\text{e}|\neg r} = 0$ reduces the results to the authentic cases from Eqs. (5.17) and (5.18) as expected (because the authentic provider has the hash point at transmission and will not make errors).

For $\mathbb{E}\left[Y_{\Delta,W}^{\text{HDSCER}}\right]$,

$$\begin{aligned}
\mathbb{E}\left[Y_{\Delta,W}^{\text{HDSCER}}\right] &= \mathbb{E}\left[Y_{\Delta}^{\text{HDSCER}}\right] \\
&= \mathbb{E}\left[g_{\Delta,\text{HDSCER}}(B_r) + N_{\Delta}\right] \\
&= \frac{1}{2r} \cdot \mathbb{E}\left[-2r + 4B_r\right] \\
&= \frac{1}{2r} \cdot \left(-2r + 4r(1 - p_{\text{e}|r})\right) \\
&= -1 + 2(1 - p_{\text{e}|r}) \; ; \\
\mathbb{E}\left[Y_{\Delta,W}^{\text{HDSCER}}\right] &= 1 - 2p_{\text{e}|r} \; . \qquad (5.71)
\end{aligned}$$

For $\mathbb{E}\left[Y_{\Sigma,W}^{\text{HDSCER}}\right]$,

$$\begin{aligned}
\mathbb{E}\left[Y_{\Sigma,W}^{\text{HDSCER}}\right] &= \mathbb{E}\left[Y_{\Sigma}^{\text{HDSCER}}\right] \qquad (5.72)\\
&= \mathbb{E}\left[g_{\Sigma,\text{HDSCER}}(B_{\neg r}) + N_{\Sigma}\right]\\
&= \frac{1}{2(n-r)} \cdot \mathbb{E}\left[2(n-r) - 4B_{\neg r}\right]\\
&= \frac{1}{2(n-r)} \cdot \big(2(n-r) - 4(n-r)p_{\text{e}|\neg r}\big) \ ;\\
\mathbb{E}\left[Y_{\Sigma,W}^{\text{HDSCER}}\right] &= 1 - 2p_{\text{e}|\neg r} \ . \qquad (5.73)
\end{aligned}$$

For $\mathbb{V}\left[Y_{\Delta,W}^{\text{HDSCER}}\right]$,

$$\begin{aligned}
\mathbb{V}\left[Y_{\Delta,W}^{\text{HDSCER}}\right] &= \frac{1}{W}\mathbb{V}\left[g_{\Delta,\text{HDSCER}}(B_{\neg r}) + N_{\Delta}\right]\\
&= \frac{1}{W}\mathbb{V}\left[g_{\Delta,\text{HDSCER}}(B_{\neg r})\right] + \frac{1}{r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W}\\
&= \frac{4}{r^2}\mathbb{V}\left[B_r\right]\frac{1}{W} + \frac{1}{r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W}\\
&= \frac{4}{r^2} r p_{\text{e}|\neg r}(1 - p_{\text{e}|r})\frac{1}{W} + \frac{1}{r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W} \ ;\\
\mathbb{V}\left[Y_{\Delta,W}^{\text{HDSCER}}\right] &= \frac{4}{r} p_{\text{e}|r}(1 - p_{\text{e}|r})\frac{1}{W} + \frac{1}{r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W} \ . \qquad (5.74)
\end{aligned}$$

For $\mathbb{V}\left[Y_{\Sigma,W}^{\text{HDSCER}}\right]$,

$$\begin{aligned}
\mathbb{V}\left[Y_{\Sigma,W}^{\text{HDSCER}}\right] &= \frac{1}{W}\mathbb{V}\left[g_{\Sigma,\text{HDSCER}}(B_{\neg r}) + N_{\Sigma}\right]\\
&= \frac{1}{W}\mathbb{V}\left[g_{\Sigma,\text{HDSCER}}(B_{\neg r})\right] + \frac{1}{n-r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W}\\
&= \frac{4}{(n-r)^2}\mathbb{V}\left[B_{\neg r}\right] + \frac{1}{n-r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W}\\
&= \frac{1}{W}\frac{4}{(n-r)^2}(n-r)p_{\text{e}|\neg r}(1 - p_{\text{e}|r}) + \frac{1}{n-r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W} \ ;\\
\mathbb{V}\left[Y_{\Sigma,W}^{\text{HDSCER}}\right] &= \frac{4}{n-r} p_{\text{e}|\neg r}(1 - p_{\text{e}|\neg r})\frac{1}{W} + \frac{1}{n-r}\frac{n}{FT}\frac{\sigma^2}{P}\frac{1}{W} \ . \qquad (5.75)
\end{aligned}$$

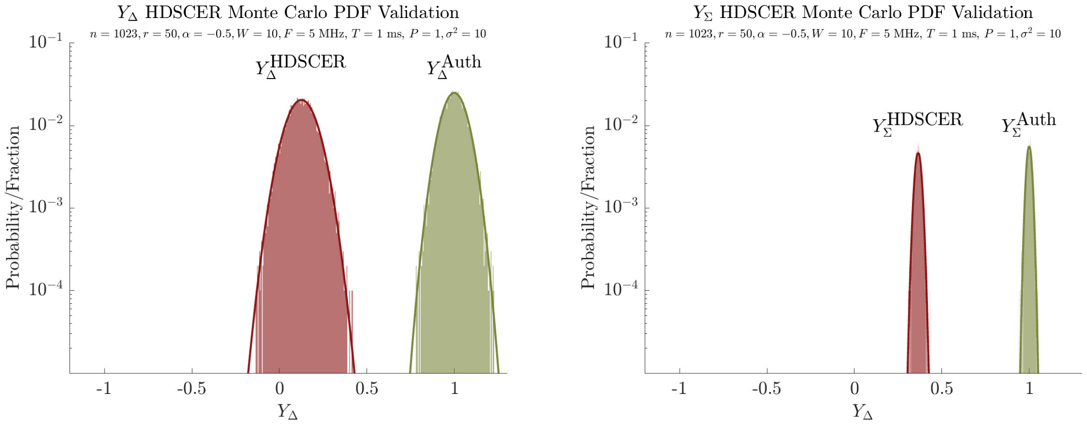


Figure 5.10: HDSCER Statistic Distribution Monte Carlo Validation.

The figure depicts the results of a Monte Carlo experiment to validate the distributions from Eqs. (5.69) and (5.70) computed via repeated convolution. The parameters were selected to show the efficacy of the distribution predictions and are $n = 1023, r = 50, \alpha = 0, P = 1, \sigma^2 = 10, F = 5$ MHz$, T = 1$ ms$, W = 10$. The experiment included 10000 trials with each trial containing $W = 10$ watermarked PRN1 ranging code with simulated noise.

### 5.4.5 Monte Carlo Validation

To validate the statistical distributions via Monte Carlo methods, I utilize the signal model from Eq. (5.21). For 10000 trials, I simulated 10 watermarked GPS PRN1 ranging codes. Each ranging code included a random watermark and simulated AWGN noise. Furthermore, each ranging code included $FT$ measurements. To simulate the adversary, the adversary observed each GNSS-generated ranging code with errors according to the BPSK model from Section 5.4.1 and transmitted a spoofed ranging code according to Section 5.4.3.

Both the authentic and the spoofed watermark signals were fed into the $Y_\Delta$ and $Y_\Sigma$ filters. For each trial, 10 $Y_\Delta$ and $Y_\Sigma$ results were averaged together to produce the data in the histogram in Fig. 5.10. The predicted authentic distributions come from Section 5.2.2. The predicted HDSCER distributions were computed via repeated convolution with Eqs. (5.69) and (5.70).

The parameters selected for simulation in Fig. 5.5 are not ideal for a real watermarking scheme. Rather, they were selected judiciously to demonstrate the efficacy of the computed distributions. For a more realistic scheme, such as that from Section 5.5 where $W = 6000$, the distributions would severely concentrate about the means, rendering such a figure uninteresting.

### 5.4.6 Expectation Trajectory and Decision Theory

Section 5.3.4 discusses the $Y_\Delta$ and $Y_\Sigma$ expectation trajectory and decision theory for the Non-SCER adversary. This section discusses the same for the HDSCER adversary. The Non-SCER is linear, resulting from Eqs. (5.40) and (5.42). For the HDSCER, the expectation trajectory is

$$\operatorname{erf}^{-1}(\mathbb{E}\left[Y_\Sigma^{\text{HDSCER}}\right]) + \operatorname{erf}^{-1}(\mathbb{E}\left[Y_\Delta^{\text{HDSCER}}\right]) = \sqrt{2\text{SNR}_{\text{SCER}}}\,. \tag{5.76}$$

I derive Eq. (5.76) in the subsection below.

Just like in Fig. 5.7, with Fig. 5.11, I derive a similar figure for the HDSCER. This time, the adversary may elect $\alpha$, the decision line on its security-chip-estimation detector.

Like with Fig. 5.7, Fig. 5.11 provides an intuitive visual demonstrating how $\alpha$ affects $Y_\Delta$ and $Y_\Sigma$. When the receiver aggregates over thousands of watermarks (e.g., $W = 6000$ from Section 5.5), the error ellipses will shrink into dots, making a less interesting figure. The watermark scheme parameters of Fig. 5.11 are not realistic to show the trend in variance over the $s$ trajectory.

The only parameter in Eq. (5.76) is the HDSCER adversary's SNR. This relates to the ***adversary's*** chip estimation equipment and signal processing ability. The better the adversary's equipment and process, the higher the adversary's SNR, the higher the level set on the expectation trajectory. This trend is depicted in Fig. 5.12.

Ultimately, as the HDSCER adversary's $p_{\text{e}|r} \to 0$ and $p_{\text{e}|\neg r} \to 0$ with better radio and computational equipment, the adversary will be able to approach perfectly estimating and replaying the watermark. The combination of the CLT distribution

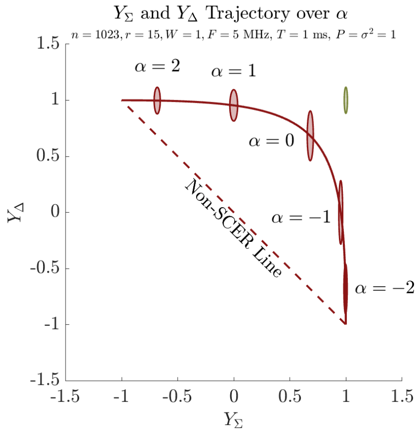


Figure 5.11: HDSCER Statistic Trajectory Over Adversarial Strategy.

The HDSCER adversary may elect any $\alpha$ strategy when attempting to break authentication security with a watermark. The figure depicts how the adversary's $\alpha$ election effects the resulting $Y_\Delta$ and $Y_\Sigma$ using the CLT simplification from Section 5.3.2. The figure depicts unrealistic noise conditions and the statistic distributions of a single watermark to demonstrate the trajectory itself and how the variances change over the trajectory. In a real scheme where the receiver averages over 1000s of watermarks (e.g., Section 5.5), the CLT effect shrinks the distributions into dots (making this figure less interesting). The authentic distribution in **green** centers at $(1, 1)$, whereas the HDSCER in **cardinal** traces a straight-line trajectory over $\alpha$.

narrowing and knowledge that a better and better HDSCER adversary could exist motivates exclusively designing based on the expectation value (and ignoring the spread of the distribution). Hence, Fig. 5.12 was based on the SNR-level sets computed via Eq. (5.76).

From a mathematical conciseness point of view, the upper tail distributions and other effects (such as advantages from Section 5.4.7) would be better accounted for by adjusting the adversary's actual SNR. For instance, rather than computing the PMDs and PFAs from the integration of the repeatedly convolved distributions of a receiver-decided decision boundary on $Y_\Delta, Y_\Sigma$, this effect could be approximated by computing the dB-width of sigma, adjusting the adversary's SNR, and continuing design entirely on these adjusted expectations. Or the SNR could be adjusted by a 3-sigma dB width from other adjustments from Section 5.4.7. To design a scheme similar to [6], it is now the problem of selecting $n$ and $r$ under an HDSCER model (i.e., how big of a dish can

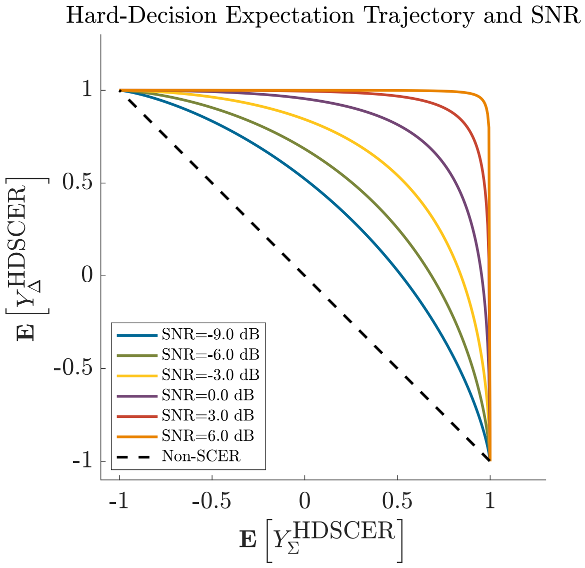


Figure 5.12: The Hard-decision Expectation Trajectory For Varying Levels Of SNR.

As the adversary's chip estimation SNR improves, the expectation trajectory moves closer to the authentication distribution centered at (1,1).

the HDSCER-capable adversary use yielding a specific SNR) that yields acceptable PMDs. However, it is possible to consider the distribution tails (rather than just the expectation) via repeated convolution of the binomial distributions.

The efficacy of receiver decisions on the $(Y_\Delta, Y_\Sigma)$ statistical space can be evaluated by integrating over the joint decision space of Fig. 5.12. In Section 5.3.4, the two statistics are conditionally independent on $h$, which serves as a single integration axis to compute the probability of missed detection. This time, $Y_\Delta, Y_\Sigma$ are conditionally independent on the hash point. This means that, from the decision's point of view (which happens after the distribution of the hash point), $Y_\Delta, Y_\Sigma$ are independent. The general independence results from how $Y_\Delta$ and $Y_\Sigma$ separate the two underlying random binomial variables. Another set of statistics would not have this advantage, motivating the selection of $Y_\Delta$ and $Y_\Sigma$.

The adversary's choice of $\alpha$ will be based on the receiver's decision line. To compute the missed detection probability, first the receiver must elect a decision line on $Y_\Delta$ and $Y_\Sigma$. Second, the receiver can perform a search on $\alpha$ to find the best PMD.

Over the search on $\alpha$, the receiver will compute the PMD leveraging that $Y_{\Delta}^{\text{HDSCER}}$ and $Y_{\Sigma}^{\text{HDSCER}}$ are independent. Therefore, assuming that the adversary elects the best $\alpha$, the receiver can determine whether the PMD is acceptable by integrating the joint, independent probability distribution $(Y_{\Delta}^{\text{HDSCER}}, Y_{\Sigma}^{\text{HDSCER}})$.

An interesting consequence of the knowledge of the $\alpha$ trajectory is a suggestion to use it as the decision line. Such a decision line may have more favorable PMDs and PFAs compared to a linear decision boundary. Note that as $Y_{\Sigma}$ decreases, the receiver's ability to track the signal rapidly decreases, informing a reasonable decision area over $Y_{\Delta}, Y_{\Sigma}$ for integration.

### Hard Decision Trajectory Equation

This section derives Eq. (5.76). First, I substitute the probability of errors with their functions of $\alpha$ from Eqs. (5.52) and (5.53) and isolate $\alpha$ for both statistics:

$$\begin{aligned}
\mathbb{E}\left[Y_{\Delta}^{\text{HDSCER}} \mid \alpha\right] &= 1 - 2p_{\text{e}|r,\alpha} \\
&= 1 - 2 \cdot \int_{\alpha}^{\infty} \text{PDF}_{\mathcal{N}(-\sqrt{P},\sigma^2)}(y)dy \\
&= 1 - 2 \cdot \Big(1 - \text{CDF}_{\mathcal{N}(-\sqrt{P},\sigma^2)}(\alpha)\Big) \\
&= -1 + 2 \cdot \text{CDF}_{\mathcal{N}(-\sqrt{P},\sigma^2)}(\alpha) \\
&= -1 + 2 \cdot \left(\frac{1}{2}\left(1 + \text{erf}\left(\frac{\alpha + \sqrt{P}}{\sigma\sqrt{2}}\right)\right)\right) \\
&= \text{erf}\left(\frac{\alpha + \sqrt{P}}{\sigma\sqrt{2}}\right) \\
\alpha &= -\sqrt{P} + \sqrt{2}\sigma \cdot \text{erf}^{-1}(\mathbb{E}\left[Y_{\Delta} \mid \alpha\right]) \, .
\end{aligned}$$

$$
\begin{aligned}
\mathbb{E}\left[Y_{\Sigma}^{\mathrm{HDSCER}} \mid \alpha\right] &= 1 - 2p_{\mathrm{e}|\neg r,\alpha} \\
&= 1 - 2 \cdot \int_{\infty}^{\alpha} \mathrm{PDF}_{\mathcal{N}(\sqrt{P},\sigma^2)}(y)dy \\
&= 1 - 2 \cdot \mathrm{CDF}_{\mathcal{N}(\sqrt{P},\sigma^2)}(\alpha) \\
&= 1 - 2 \cdot \left(\frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{\alpha - \sqrt{P}}{\sigma\sqrt{2}}\right)\right)\right) \\
&= -\,\mathrm{erf}\left(\frac{\alpha - \sqrt{P}}{\sigma\sqrt{2}}\right) \\
&= \sqrt{P} + \sqrt{2}\sigma \cdot \mathrm{erf}^{-1}(-\mathbb{E}\left[Y_{\Sigma} \mid \alpha\right]) \\
\alpha &= \sqrt{P} - \sqrt{2}\sigma \cdot \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Sigma} \mid \alpha\right]) \ .
\end{aligned}
$$

Then, I set the $\alpha$ equal to each other:

$$
\begin{aligned}
\sqrt{P} - \sqrt{2}\sigma \cdot \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Sigma} \mid \alpha\right]) &= -\sqrt{P} + \sqrt{2}\sigma \cdot \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Delta} \mid \alpha\right]) \\
\sqrt{2}\sigma \cdot \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Sigma} \mid \alpha\right]) + \sqrt{2}\sigma \cdot \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Delta} \mid \alpha\right]) &= 2\sqrt{P} \\
\mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Sigma} \mid \alpha\right]) + \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Delta} \mid \alpha\right]) &= \sqrt{2P/\sigma^2} \\
\mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Sigma} \mid \alpha\right]) + \mathrm{erf}^{-1}(\mathbb{E}\left[Y_{\Delta} \mid \alpha\right]) &= \sqrt{2\mathrm{SNR}_{\mathrm{SCER}}} \ .
\end{aligned}
$$

Note that the SNR here is the SNR of the HDSCER adversary, which is a function of the adversary's radio and signal processing equipment.

### 5.4.7 Soft Information Advantage

In the hard-decision adversary from Section 5.4.3, the adversary makes a hard decision on the security code estimation problem. This ignores potentially useful soft information, such as the measurement likelihood from the BPSK model. Moreover, the hard-decision adversary employs a constant chip power.

In this section, I discuss two attempts to create an adversary that leverages soft information to beat the distributions predicted for the HDSCER. In the first attempt, I observed no advantage, and I provide reasoning as to why. In the second attempt, I allowed the adversary to transmit varying powers per chip. The adversary incorporates soft information into the individual chip powers. I call this adversary the PSCER adversary.

### Soft Information First Attempt

Each time the adversary makes a chip estimation measurement, it can utilize the model provided in Section 5.4.1 to generate soft information. The BPSK measurement model from Fig. 5.9 provides a measurement likelihood.

For the HDSCER adversary, the adversary makes independent chip measurements and makes inversions without knowledge of the $r$ structure of the watermark. As a degenerate case, the adversary could, based on its chip measurements, invert every single chip in a watermark. Instead, suppose the adversary inverted the $s$ chips with the highest measured inverted likelihoods for its transmitted spoof. For instance, it could invert the $s = r$ chips with the $r$ highest measured likelihoods of being inverted. While a reasonable thing to do, I observe no advantage.

This strategy reveals that the $\alpha$ decision parameter for the HDSCER is an analog to $s$ for the Non-SCER. As $p_{\mathrm{e}|r}$ and $p_{\mathrm{e}|\neg r}$ approach 0.5, when the chip estimation is useless, the $\alpha$ trajectory from Section 5.4.3 approaches the $s$ trajectory for the Non-SCER adversary. To show correspondence, I derive the expected number of inversions $\mathbb{E}[s]$ given an $\alpha$:

$$\begin{aligned}\mathbb{E}[s \mid \alpha] &= r \cdot \mathbb{E}[s \mid \alpha, r] + (n-r) \cdot \mathbb{E}[s \mid \alpha, \neg r] \\ \mathbb{E}[s \mid \alpha] &= r(1 - p_{\mathrm{e}|r}) + (n-r)p_{\mathrm{e}|\neg r} \ . \end{aligned} \tag{5.77}$$

Using this correspondence, I generate Fig. 5.13 with varying $\alpha$ and $s$ with colors indicating correspondence. As the HDSCER SNR approaches $-\infty$, the lines and colored dots converge.

Using Monte Carlo experimentation, I found that the HDSCER adversary that selects the $s$ chips with the measured highest likelihoods of being inverted produces a strategy on the HDSCER trajectory line. While Eq. (5.77) computes the expected $s$ from $\alpha$, when the adversary selects $s$, it enacts an expected $\alpha$. Therefore, this strategy

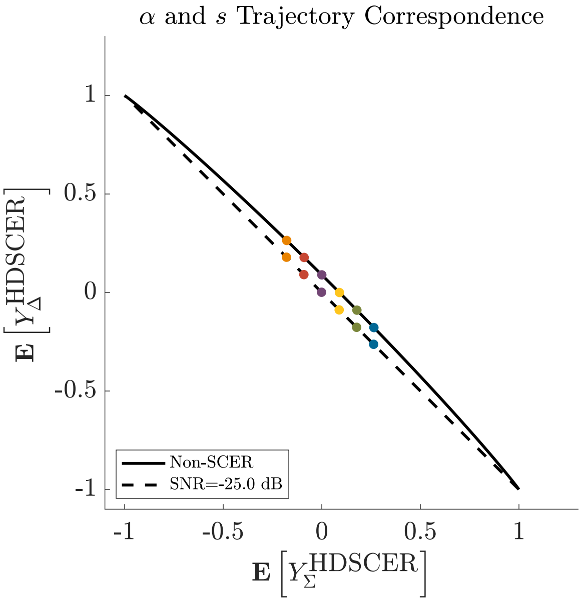


Figure 5.13: Correspondence Of HDSCER And Non-SCER Adversary.

The figure generates the HDSCER trajectory with a very low SNR to show that the HDSCER trajectory approaches the Non-SCER adversary as the HDSCER chip estimation approaches useless. The HDSCER is set with some SNR to ensure the lines are seperated for visual purposes. Using Eq. (5.77), a varying set of $\alpha$ and $s$ are generated with colors indicating coorespondence.

is just another way of selecting $\alpha$, and the strategy produces results on the HDSCER trajectory line. To show an advantage, the strategy needs to be on the right side of the trajectory line, closer to the authentic distribution centered at $(1, 1)$. However, using this strategy always produced a distribution on the HDSCER expectation trajectory.

### PSCER Advantage

In [75], the authors present watermarking statistics and note that the adversary can induce an individual power on each chip. In this section, I construct an adversary that incorporates soft information into the spoof by relating the power of each chip to the measured certainty of the chip estimation. I call this adversary the Power-SCER adversary, or PSCER, without any claim about whether this adversary is the best obtainable.

With PSCER, the adversary will set the chip power to be proportional to the hypothesis likelihood:

$$P_i \propto \begin{cases} \Pr(r \mid S_i) & \Pr(r \mid S_i) > \Pr(\neg r \mid S_i) \\ \Pr(\neg r \mid S_i) & \text{otherwise} \; . \end{cases} \tag{5.78}$$

The hypothesis value comes from the BPSK model from Section 5.4.1. Again, the adversary may elect an $\alpha$, which relates to the adversary's election of the two chip hypotheses' prior probabilities. PSCER will invert the chips when the measured likelihood of inversion is larger than that of non-inversion.

My choice for $P_i$ is simply a judicious, first-guess choice inspired by [75] that serves my intuitive purpose. When the adversary is very confident that a chip is inverted, it will place more power on that chip (and the same with a chip highly believed not to be inverted). When the adversary is not confident that a chip is inverted or not inverted, the adversary places less power on that particular chip. For the PSCER adversary, I re-normalize the signal to contain the same aggregate average power over the entire ranging code; hence, I use $\propto$ for Eq. (5.78). This accounts for tracking loop automated gain control and establishes a fair comparison in the $Y_\Delta$ and $Y_\Sigma$ space. I tried a couple of other functions that ensure more power on more confidence measurements (e.g., having $P_i$ be a function of the likelihood ratio) with varying advantages. But I present the simplest one for this work.

Fig. 5.14 depicts the PSCER $\alpha$ trajectory measured via Monte Carlo simulation. For five $\alpha$ values, I conducted 100 trials of $W = 6000$ watermark aggregations. Because of the $W = 6000$, the spread of the resultant distribution of $Y$s is concentrated. However, for each $\alpha$, the distribution of $Y$s fall to the right of the HDSCER $\alpha$ trajectory. Because the Monte Carlo distributions are to the right of the HDSCER $\alpha$ trajectory, PSCER demonstrates an advantage over HDSCER.

The parameters from Fig. 5.14 follow Section 5.5 for realism purposes. Even with realistic watermark and noise parameters, the advantage is small, which reflects the relative gains with soft information in error correction code contexts.

At the time of this work, I do not yet observe a pathway to mathematically derive

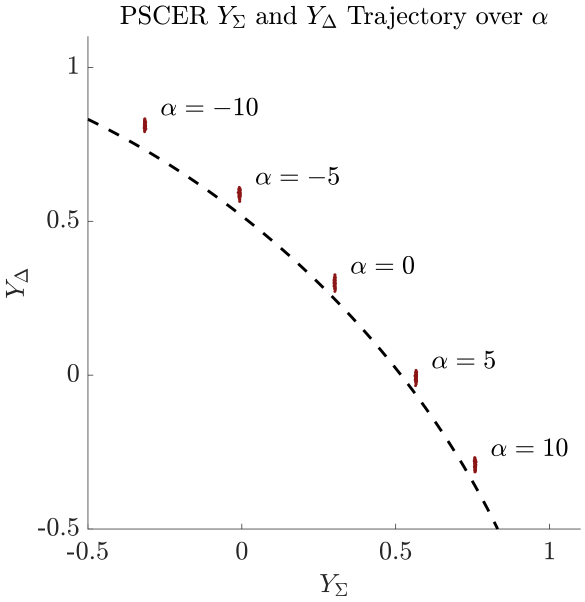


Figure 5.14: Monte Carlo Experiment Showing PSCER Advantage.

From Section 5.5, $n = 1023$, $r = 15$, $\frac{C}{N_0} = 47$ dB-Hz, $F = 5$ MHz, $T = 1$ ms, $W = 6000$. Five $\alpha$ values, with 100 trials, are depicted along with the HDSCER $\alpha$ trajectory line. Since Monte Carlo distributions are to the right and above the HDSCER close to the authentic distribution centered at $(1, 1)$, the PSCER adversary has an advantage over the HDSCER adversary.

the advantage for PSCER or other soft-decision adversaries, to find the best soft-decision adversary, and bound the advantage of any soft-decision adversary. Given the convenience of the mathematically concise derivations for the hard-decision adversary and the conventions of error-correction code, it is likely appropriate to attempt to find a soft-information advantage bound or correction for use in designing a system with the hard-decision derivations. For instance, suppose one could show that a soft-decision adversary performs no better than a hard-decision adversary with $x$ more SNR dB. Then, one could design using the hard decision formulae with simple corrections.

Because an adversary could continually achieve a better radio for the security code estimation, the GNSS designer should focus on ensuring that the system design requires an antenna that is reasonably arduous on the spoofer and easy for someone in the area to detect. For instance, one could design the system to require a large dish antenna that would likely be visible in a protection area (e.g., in the vicinity of an airport). Noting that the $r = 15$ design from [6] was created before this work, one can derive the gain required to spoof a receiver. [6] suggested a decision boundary of $Y_\Delta > 0.5$ for $10^{-9}$ missed-detection and false-alarm rates for Non-SCER adversaries. To spoof a receiver ***in expectation***, the adversary would need an antennae array or a high-gain antenna until the spoofing ellipses from Fig. 5.14 cross past the receiver's decision boundary (e.g., $Y_\Delta, Y_\Sigma > 0.5$).

Deriving a rigorous answer to the advantage of a soft-decision adversary poses a difficult challenge for both deriving an answer and defining a model. For instance, in the model of this work, an adversary could put an enormous power on a single chip (and zero out the other chips). Among the entire ranging code measurements, suppose the adversary only placed power on two chips: the one with the highest measured likelihood of being inverted and the one with the highest measured likelihood of not being inverted. Likely, these two measurements (e.g., among the $r$ and $n - r$) are correct. With a perfectly tracking receiver, the adversary could spoof $Y_\Delta$ and $Y_\Sigma$ by placing max power on those two chips. However, this represents a degenerate case, motivating a more sophisticated receiver and spoofing radio models (e.g., where the power of these chips is saturated in the 2-chip power spoof). As the model becomes more complicated and realistic, a mathematically concise answer is unlikely, relegating the best answer to Monte Carlo methods and direct experimentation.

## 5.5 Application to SBAS

In this section, I apply the methods of the previous sections to design a watermark for WAAS: $n = 1023$, $r = 4$, $W = 6000$ for 32-bit security. This watermark will induce a 0.03 dB loss on the signal and adopt the SBAS six-second cadence from Section 3.4.

While the watermark presented in this section may be applied to any SBAS, many SBAS, such as European Geostationary Navigation Overlay Service (EGNOS) do not support ranging. Since WAAS does support ranging, this scheme design applies specifically to WAAS and any other SBAS that supports ranging. Moreover, since

WAAS signals are generated on the ground, WAAS presents possibly the quickest avenue for implementing a public GNSS-ranging authentication scheme.

Section 5.5.1 discusses the needed cryptography construction assuming the SBAS TESLA scheme from Section 3.4. Sections 5.5.2 and 5.5.3 discuss how I selected the SBAS watermark parameters heuristically and the underlying noise and security assumptions. Section 5.5.4 computes the watermark PMD and PFAs via repeated convolution to verify the security requirements, and Section 5.5.5 discusses SBAS watermark validation experiments. And finally, Section 5.5.6 briefly discusses how the results of this section can be quickly adapted by any GNSS.

### 5.5.1 Cryptographic Construction

Recall from Section 3.4 and [13] that SBAS TESLA will distribute a hash point every six seconds. In support of an SBAS watermarking scheme that minimally requires additional design changes, it would be apt to provide a watermark at that cadence. And to minimize the amount of degradation on the SBAS ranging code while utilizing the natural millisecond divisions provided by the ranging code, each C/A ranging code will receive its own watermark. Therefore $n = 1023$ and $W = 6000$.

To exploit TESLA's data efficiency property to authenticate yet another object in the signal, I introduce another branch of the hash path via KDF. To support an independent watermark for each ranging code, 6000 additional KDF operations per hash point are needed to generate 6000 watermark seeds to be utilized with the inverted-chip-selection Algorithm 5.1. This amounts to 1000 per second. This could be formulated by

$$p_{w,t_j,t_{\mathrm{ms}}} = \mathrm{KDF}(p_{t_j}, \text{“Watermark Seed”}||\mathrm{PRN}||\mathrm{Frequency}||t_j||t_{\mathrm{ms}}) \ . \tag{5.79}$$

In Eq. (5.79), the string “Watermark Seed” ensures a unique context to any other KDF derivations pertaining to which satellite and frequency. $t_j||t_{\mathrm{ms}}$ ensures a unique context per watermark for all time. From $p_{w,t_j,t_{\mathrm{ms}}}$, one derives the actual chips inverted from Algorithm 5.1. Fig. 5.15 depicts the cryptographic geometry of this construction.

With this construction, as depicted in Fig. 5.15, the TTA will be 12 seconds, and

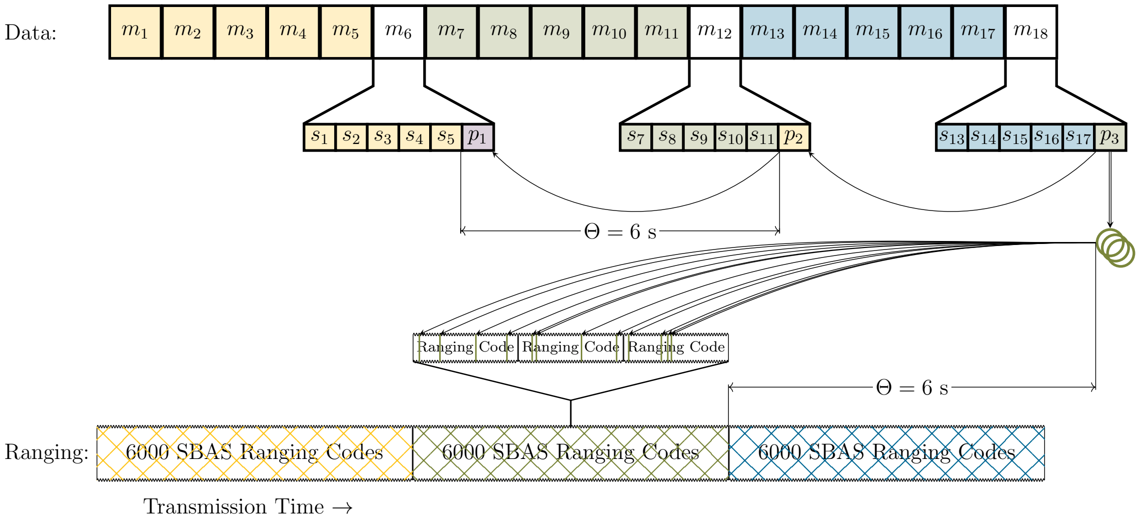

Data:
$m_1$ $m_2$ $m_3$ $m_4$ $m_5$ $m_6$ $m_7$ $m_8$ $m_9$ $m_{10}$ $m_{11}$ $m_{12}$ $m_{13}$ $m_{14}$ $m_{15}$ $m_{16}$ $m_{17}$ $m_{18}$
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $p_1$
$s_7$ $s_8$ $s_9$ $s_{10}$ $s_{11}$ $p_2$
$s_{13}$ $s_{14}$ $s_{15}$ $s_{16}$ $s_{17}$ $p_3$
$\Theta = 6$ s
Ranging Code Ranging Code Ranging Code
$\Theta = 6$ s
Ranging:
6000 SBAS Ranging Codes
6000 SBAS Ranging Codes
6000 SBAS Ranging Codes
Transmission Time →

Figure 5.15: Conceptual Diagram Of The Proposed SBAS Watermark.

The diagram depicts the cryptographic construction of the SBAS watermark from this section ($n = 1023, r = 4, W = 6000$). The watermark poses no burden on the SBAS data bandwidth since it derives from the TESLA hash points. The colors correspond to which watermarks are derived from which hash point delivered via MT50 every six seconds. Because hash points are distributed in the middle of a message, the groups of 6000 watermarked SBAS ranging codes do not start and end in alignment with the start and end of data messages. Instead they align with the distribution of the hash points to minimize TTA while respecting the TESLA commitment reveal delay ($\Theta$).

the required TESLA time synchronization is $\Theta = 6$ seconds is concurrent with SBAS NMA. The colors in Fig. 5.15 correspond to which hash point derives which ranging code's inverted chips.

Because $2^{\lceil \log_2 n \rceil} = 1024$, the chip drawing algorithm will need ten pseudorandom bits per integer drawn. The later sections will derive $r = 4$, meaning that 40 bits will be needed. If HMAC-SHA256 is used for KDF, this means a single HMAC per watermark chip drawing. Together with the KDF to get $p_{w,t_j,t_{\text{ms}}}$, and noting that each HMAC needs two SHA256 calls, this amounts to 4 hashes per watermark and 4000 hashes per second. While the hashing rate on my laptop exceeds this number by several orders of magnitude, a GPU would enable enormous hashing rates and parallelization of this operation.

### 5.5.2 Design with the CLT and Parallel Decision Lines

Because a better and better SCER adversary will eventually break any watermark, I will first design a scheme that meets the 32-bit security requirement for the Non-SCER adversary. Then, in Section 5.5.4, I will predict the required SCER radio equipment needed to break the scheme to advise authorities on what equipment to look for near aircraft transit areas.

When designing watermark parameters (e.g., $n$, $r$, $W$), the final objective will be the PMD and PFA on the decision space of $Y_\Delta$ and $Y_\Sigma$. The main design levers to trade off are $r$ and $W$. Ideally, $r$ should be minimized to limit the signal's degradation, and $W$ should be minimized to limit the receiver's memory storage needed for watermark signal processing. To conduct this optimization, one must search among the scheme parameters: $r$, $W$, $\frac{C}{N_0}$, PMD, $F$, and any decision line.

A search among $r$, $W$, $\frac{C}{N_0}$, PMD, $F$, and the decision line using the distributions computed via convolution would be an arduous affair. Because of $W$'s size, the closed-form CLT approximations from Section 5.3.2 can assist with design. I suggest designing a watermark using the CLT approximations to find an initial starting point. From the starting point, one can then tweak the parameters as in Section 5.5.3. After that, one should then use the convolution-generated distributions to verify the selected watermark meets the PMD and PFA requirements as in Section 5.5.4. For the remaining part of this section, I will use the CLT approximations from Section 5.3.2 to find a watermark with a dispositive decision problem.

I assume that an aviation receiver will refuse to track a GNSS signal with a $\frac{C}{N_0} < 30$ dB-Hz and will have a radio better than $F = 2$ MHz. These noise and radio parameters are meant to provide an unreasonable lower bound for conservative security design since signals are typically above $\frac{C}{N_0} > 40$ and the needed Nyquist frequency is 2.046 MHz.

First, I suggest electing a decision line parallel to the $s$ trajectory. The PMD and PFA will be based on the center and spread of the $Y_\Delta$ and $Y_\Sigma$ statistics. Electing a decision line parallel to the $s$ trajectory diminishes the effect of the distribution ***centers*** on the PMD. The effect of the spreads is still there and changes over the $s$ trajectory as observed in Fig. 5.7. In addition, I suggest a decision line halfway between the $s$ trajectory and the authentic center at $(1, 1)$: $Y_\Delta + Y_\Sigma = 1$. Having this decision line makes the effects of the spreads on the PMD and PFA approximately the same, which is convenient if one wants the PMD to be approximately the same as the PFA. I desire to match the 32-bit ($10^{-9}$) security level afforded to SBAS NMA for both the PMD and the PFA.

The left diagram in Fig. 5.16 captures the intuition of the following design process. In **green**, Fig. 5.16 depicts the authentic hypothesis centered at $(1, 1)$ and in **cardinal** the Non-SCER hypotheses for varying $s$. With the parallel decision line $Y_\Delta + Y_\Sigma = 1$, the worst-case spread (which results in the worst PMD) occurs when $s = 511$ and $s = 512$. So, from here, one can design assuming the worst-case $s$.

With just the worst-case $s$, the decision problem returns to the standard two Gaussian hypotheses decision problem. Since the decision line is $Y_\Delta + Y_\Sigma$, it makes sense to compute the cumulative density functions along the $Y_\Delta = Y_\Sigma$ line, which is done in the right plot of Fig. 5.16 for varying $W$.

From Section 5.5.1, $n = 1023$ and $W = 6000$, but for the moment, I will ignore $W = 6000$ to show how this design procedure works for other GNSS. In Eq. (5.41), when $n, s, F, T, P$, and $\sigma^2$ are held constant, $\mathbb{V}\left[Y_\Delta^{\neg\text{SCER}}\right]$ scales equally with $r$ and $W$: $\frac{1}{r}$ and $\frac{1}{W}$. This means that after finding an $r \cdot W$ that meets the PMD requirement, one can approximately inversely trade $r$ and $W$.

While $W$ can be increased substantially (until the TTA is unacceptable), $r$ cannot because there are only $n$ chips and substantially increasing $r$ risks making tracking more difficult. Fig. 5.16 performs a search on $r = 100$ arbitrarily to show that this trade-off behavior prediction works. To meet the design requirement, from Fig. 5.16, $r \cdot W = 23000$. Recalling from Section 5.5.1, where $W = 6000$, this predicts that $r = 4$ chips are required. With $r = 4$ as a starting point, one can now perturb the system parameters to understand their effects in this parameter optimization problem.

Fig. 5.17 provides the results to the following. Given $r$, what is the minimum $W$ that will satisfy a specific PMD:

$$\underset{W}{\operatorname{argmin}} \Pr(Y_\Delta + Y_\Sigma \geq 1 \mid \neg\text{NSCER}, n, r, s, P, \sigma^2, F, T, W)\ . \tag{5.80}$$

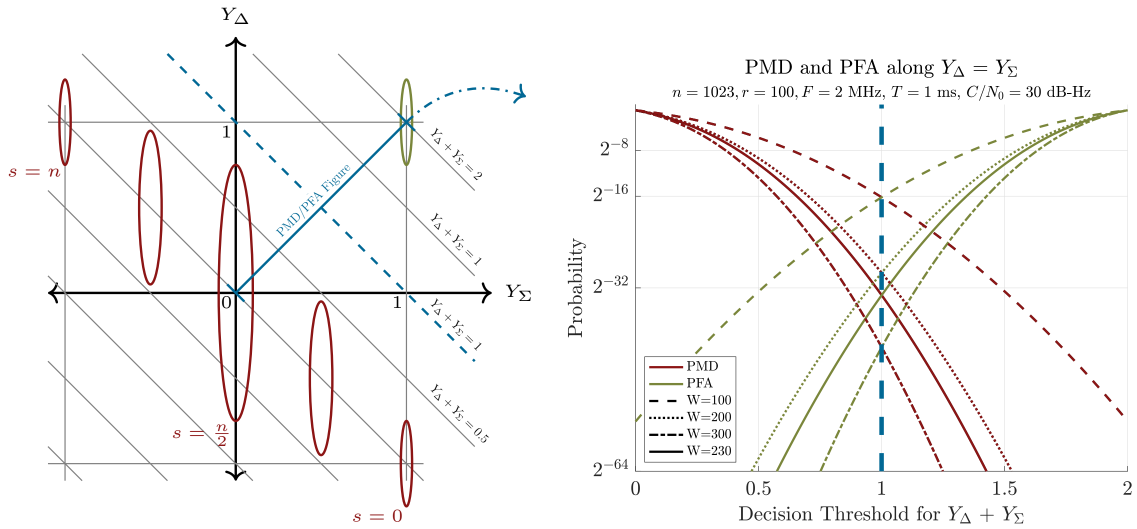


Figure 5.16: Non-SCER Watermark Design With CLT Approximation.

On the left shows an intuitive distribution contour plot of the authentic (**green**) and Non-SCER (**cardinal**) hypotheses. If this diagram were a real watermark supporting 32-bit security, the ellipses would be around the 6-$\sigma$ ellipses. A decision line parallel to the $s$ trajectory eliminates the effect of the moving non-SCER distribution center in the PMD calculation, leaving only the spread to account. The largest spread occurs when $s = 511$ and $s = 512$ when $n = 1023$. Using the properties of sums of Gaussian distributions, one can have the PMD and PFA integration axis be orthogonal to the decision line. On the right, the PMD and PFA are calculated along this axis for varying $W$, to find the $W$ that produces the requested PMD and PFA. As discussed in Section 5.5.2, from the diagram, the watermark scheme will need to have $r \cdot W \approx 23000$, shown with solid lines, to meet a 32-bit security requirement. For other $s$, the spread will be smaller, meaning the PMD will be better.

Luckily, this $W$ is continuously decreasing with increasing $r$, meaning that to find the minimum $W$, one can do a binary search, as was done with Fig. 5.17 on each reported $r$ in about an hour with a laptop. By computing the PMD from the distribution computed via convolution, Fig. 5.17 verifies that $r$ and $W$ are inversely related with constant PMD, but not perfectly so.

When creating Fig. 5.17, one could swap the roles of $r$ and $W$ in Eq. (5.80). That is, compute the minimum $r$ given $W$. However, since the $r$ varies integrally when $W$ varies by the 1000s, creating a smooth figure like Fig. 5.17 would require evaluating 1000s of $W$ (rather than the fewer $r$ needed in Fig. 5.17).

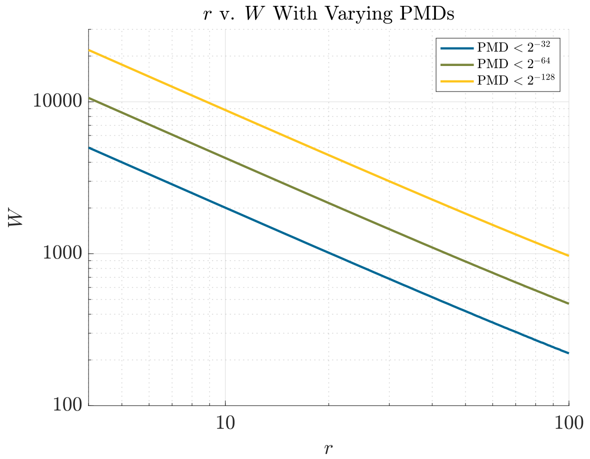


Figure 5.17: $r$ v. $W$ Over Varying PMDs.

### 5.5.3 Perturbing Model Parameters for SBAS

This section perturbs the other system parameters from the starting point provided by Section 5.5.2. The general approach is to examine trends when perturbing some system parameters while keeping others constant. This is analogous to solving a non-convex problem by (1) initializing a starting point and (2) linearizing the objective to scope around in certain directions. As a matter of practicality, I have three axes to make perturbations: abscissa, ordinate, and color. Switching the abscissa and ordinate does not really generate a new trend plot. Therefore, there are five parameters to examine: $r$, $W$, $\frac{C}{N_0}$, PMD, and $F$, and there are $5 \cdot 4 \cdot 3/2 = 30$ possible trend plots to consider.

To start, I remove $F$ from this examination. Among reasonable mass-produced receivers, 2 MHz $< F <$ 20 MHz. Compared to perturbing other parameters, across that $F$ range, there is not much effect on the scheme overall. $F$ analogously affects $\frac{C}{N_0}$: the larger the $F$ the lower the $\frac{C}{N_0}$ the receiver can tolerate. But the applicable $\frac{C}{N_0}$ is vast and therefore represented with a log scale. It also seems unwise to specify a sampling rate needed for authentication with receivers, given the potential need for a highly accurate clock to maintain a higher-frequency sampling rate. Therefore, $F = 2$ MHz is eliminated from this perturbation study, leaving $4 \cdot 3 \cdot 2/2 = 12$ possible trend plots to consider.

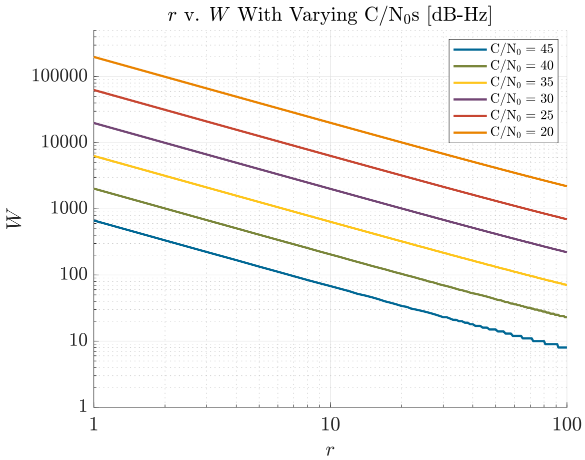


Figure 5.18: $r$ v. $W$ Over Varying C/N$_0$s.

Next, I set the PMD to the 32-bit level. Since 32-bit security is sufficient for NMA, 32-bit watermarking security should also be sufficient for the watermark, leaving $3 \cdot 2 \cdot 1/2 = 3$ possible trend plots to consider. The following paragraphs cover these three remaining graphs. For each case, the linear lines reflect the prediction from the CLT formulations from Section 5.3.2 but are computed using the distributions computed via convolution.

Fig. 5.18 plots $r$ against $W$ for varying $\frac{C}{N_0}$. This plot was generated from the convolution-computed distributions and via Eq. (5.80), except with 32-bit security and varying $\frac{C}{N_0}$.

Fig. 5.19 plots $r$ against $\frac{C}{N_0}$ for varying $W$. This plot was generated with a binary search solving

$$\operatorname*{argmin}_{\frac{C}{N_0}} \Pr(Y_\Delta + Y_\Sigma \geq 1 \mid \neg\text{NSCER}, n, r, s, P, \sigma^2, F, T, W)\ . \tag{5.81}$$

Fig. 5.20 plots $W$ against $\frac{C}{N_0}$ for varying $r$. This plot was generated by solving Eq. (5.81).

The plots from Figs. 5.18 to 5.20 are very log-inverse-linear. These all reflect the formulations from the inverse relationship within the CLT variances with constant PMD. Because of these linear relationships, the GNSS designer can be confident

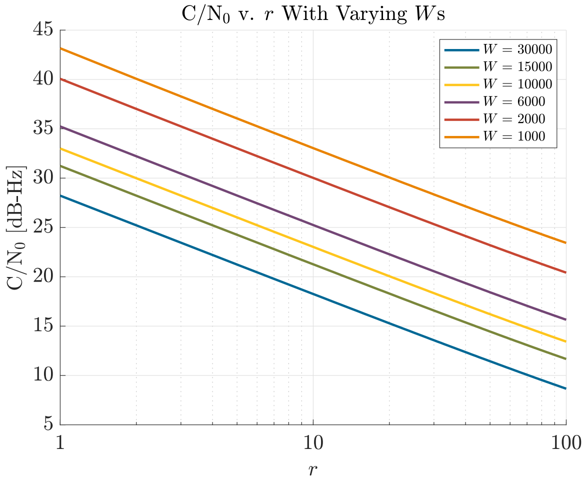


Figure 5.19: $r$ v. C/N$_0$ Over Varying $W$s.

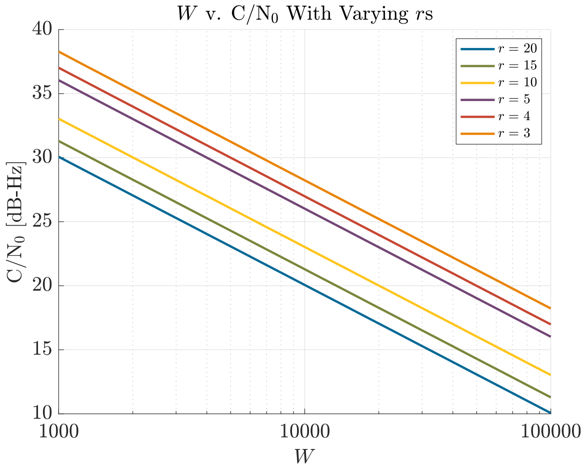


Figure 5.20: $W$ v. C/N$_0$ Over Varying PMDs.

that utilizing the CLT formulation should provide an excellent initial design study to explore the full parameter space. After finding acceptable parameters, the last step is to verify the PMD requirement, as discussed in the following section.

### 5.5.4 Watermark Security Evaluation

This section evaluates the proposed (i.e., $n = 1023$, $r = 4$, $W = 6000$) SBAS watermark's final PMD and PFA to ensure it meets the intended requirements. The

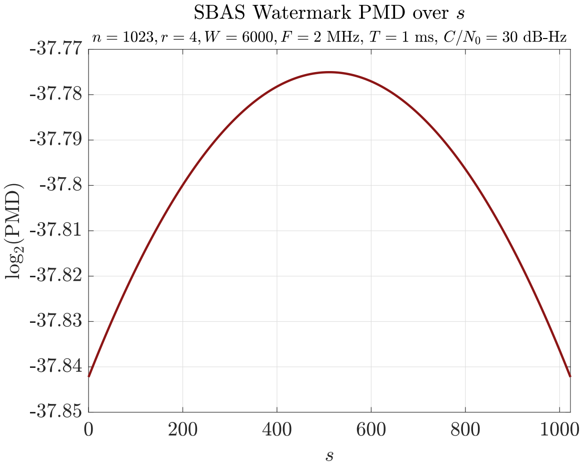


Figure 5.21: PMD Over $s$ For The Proposed SBAS Watermark.

analysis in Sections 5.5.2 and 5.5.3 is heuristic. So, the final analysis of this section is needed after arriving at a final scheme.

Fig. 5.21 computes the $\log_2(\text{PMD})$ over the Non-SCER adversary's election $s$ via Eq. (5.49). Since for every $s$ the PMD is less than $2^{-32}$, the watermark affords 32-bit security and meets the design requirement. The computed PFA is less than $2^{-37.8}$, which nicely matches the PMD, reflecting the intention from Section 5.5.2., Therefore, this watermark design meets both the PMD and the PFA requirements.

A better and better HDSCER will eventually be able to break the watermarking scheme. Such SCER adversaries require significant technical expertise, so meeting 32-bit security against the Non-SCER adversary may be sufficient for SBAS. Rather than designing the watermark against SCER adversaries, I think the right course is to determine the necessary SCER chip estimation equipment gain. With that gain, authorities in sensitive areas (e.g., aircraft transit areas) can look for this equipment.

Given the low false alarm rate, I recommend that when an alarm is thrown, the receiver ignores GNSS for some time. Then, for an SCER adversary to be successful, it will need to successfully spoof consecutive watermarked ranging codes for the length of the spoof. For the SCER adversary to do this, it should produce the needed spoofs more often than not. In the context that $W = 6000$, this means the adversary must ensure that $\mathbb{E}\left[Y_{\Delta,W}^{\text{SCER}} + Y_{\Sigma,W}^{\text{SCER}}\right] \geq 1$. And the HDSCER trajectory Eq. (5.76) relates this expectation to the adversary's chip estimation SNR.

The HDSCER trajectory from Eq. (5.76) is symmetric about $Y_\Delta$ and $Y_\Sigma$. And since the decision line is also symmetric over $Y_\Delta$ and $Y_\Sigma$, the point where $\mathbb{E}\left[Y^{\text{SCER}}_{\Delta,W} + Y^{\text{SCER}}_{\Sigma,W}\right] \geq 1$ will be the SNR at which Eq. (5.76) is tangent to the decision line. This tangent point will be on the line $Y_\Delta = Y_\Sigma$ line when the HDSCER adversary's threshold $\alpha = 0$ and $p_{\text{e}|r} = p_{\text{e}|\neg r}$. Therefore, starting from Eq. (5.76):

$$\begin{aligned} \text{erf}^{-1}(\mathbb{E}\left[Y^{\text{HDSCER}}_{\Delta}\right]) + \text{erf}^{-1}(\mathbb{E}\left[Y^{\text{HDSCER}}_{\Sigma}\right]) &= \sqrt{2\text{SNR}_{\text{SCER}}} && (5.76) \\ \text{erf}^{-1}(0.5) + \text{erf}^{-1}(0.5) &= \sqrt{2\text{SNR}_{\text{SCER}}} \\ \text{SNR}_{\text{SCER}} &= 0.454 = -3.43\text{dB}\ . && (5.82) \end{aligned}$$

Assuming that the HDSCER is observing chips ideally with a C/N$_0$ = 50 dB-Hz, using Eq. (5.20), and assuming the adversary's sampling rate is 2 MHz, this corresponds to a pre-correlation SNR of -10 dB. A real SCER adversary will need more than 2 MHz, decreasing the pre-correlation SNR. Therefore, the adversary will need at least a 6.57 dB gain antennae.

To account for the PSCER adversary from Section 5.4.7, Fig. 5.11 shows an improvement of about 0.03 in the $Y_\Delta = Y_\Sigma$ direction when $\alpha = 0$. One can adjust the decision boundary to 0.47 to account for the PSCER advantage over HDSCER. Redoing the above procedure with a decision boundary of $Y_\Delta + Y_\Sigma = 0.47$ predicts the need for a 6 dB gain antennae.

While this section's analysis stops here, some improvements could still make the SCER resistance better. For instance, the linear decision boundary could be replaced because the PFA is so low. I would first investigate a decision boundary in the shape Eq. (5.76). Additionally, $r$ or $W$ could be increased.

### 5.5.5 Watermark Experimental Validation

In this section, I provide experimental evidence that the distributions from Section 5.3.1 are correct for the SBAS watermark of this section. First, I discuss a Monte Carlo simulation. Second, I discuss experimentation with an SDR with WAAS data.

### Monte Carlo Validation

To validate Eq. (5.49), I conducted a series of Monte Carlo experiments. With the SBAS watermark proposed, I should expect a missed detection or false alarm every $6 \cdot 2^{32}$ seconds. Observing such an event is outside the computational resources available to me. Therefore, to observe these events, I simulate with a much lower $\mathrm{C/N_0}$ to show that derived probabilities are predictive.

For each Monte Carlo experiment, each with a different simulated $\mathrm{C/N_0}$, the following was repeated $10^6$ times. The simulation generated 6000 WAAS PRN 131 ranging codes and randomly inverted $r = 4$ chips in each ranging code. The simulated adversary inverted $s = 511$. The original ranging code, watermarked ranging code, and adversary-generated ranging code were then resampled to have 2000 samples each millisecond ($F = 2$ MHz) to create $R$, $R^w$, $R^{\neg\mathrm{SCER}}$, respectively. Then AWGN noise was incorporated into simulated signals from which $Y_\Delta$ and $Y_\Sigma$ were compared against the $Y_\Delta + Y_\Sigma = 1$ threshold. A single experiment with $10^6$ trials, each with 6000 ranging codes, corresponding to $6 \cdot 10^9$ ms or about 70 days of ranging code, took about one day in real-time on a Matlab instance with 16 parallel computing workers.

Table 5.2 provides the Monte Carlo results for each of the simulated noise conditions. In each case, the Monte Carlo result fell within the CLT 3-sigma (99.7%) confidence bounds of the predicted probability. My selection of $\mathrm{C/N_0}$ for experiments was to ensure some observation of rare events and show the efficacy of the model in support of designs assuming a trusted, worst-case $\mathrm{C/N_0}$ of 30 dB-Hz. In the case with $\mathrm{C/N_0}$ of 30 dB-Hz, the original scheme prediction stands, which has adverse-event probabilities less than $2^{32} < 10^{-9}$, predicting no missed detections or false alarms expected within $10^6$ trials.

### Validation with WAAS SDR

For this experiment, I reuse the mirrored concept from Section 5.3.3. However, this time it is applied to WAAS SDR data and the watermark is $r = 4, W = 100$. The data was taken on July 12, 2023, at 15.4 MHz with a USRP N310 over one hour. I used a value of $W = 100$ rather than the proposed $W = 6000$ to have more samples

| **C/N$_0$** [dB-Hz] | **Monte Carlo PFA** | **Eq. (5.51) PFA** | **Monte Carlo PMD** | **Eq. (5.50) PMD** |
|---|---|---|---|---|
| 30 | 0 | $4 \cdot 10^{-12} \pm 6 \cdot 10^{-9}$ | 0 | $4 \cdot 10^{-12} \pm 6 \cdot 10^{-9}$ |
| 25 | 0.00006 | $0.00006 \pm 0.00002$ | 0.00007 | $0.00006 \pm 0.00002$ |
| 20 | 0.01510 | $0.01531 \pm 0.00037$ | 0.01531 | $0.01532 \pm 0.00037$ |
| 15 | 0.11213 | $0.11205 \pm 0.00095$ | 0.11210 | $0.11205 \pm 0.00095$ |
| 10 | 0.24697 | $0.24710 \pm 0.00129$ | 0.24750 | $0.24710 \pm 0.00129$ |

Table 5.2: Monte Carlo Results To Validate SBAS Watermark Security.

Results from multiple Monte Carlo simulations to validate the models of Eqs. (5.49) and (5.51) under $n = 1023$, $r = 4$, $s = 511$, $F = 2$ MHz, $T = 1$ ms, $W = 6000$. The confidence intervals reported are the 3-sigma (99.7%) intervals after applying the CLT assuming the experimental results follow a binomial distribution using the PMD and the PFA. With C/N$_0$ of 30, the experiment returns to the proposed watermark scheme with adverse event probabilities less than the $2^{-32} < 10^{-9}$ requirements. And given that I simulated each situation $10^6$ times, I expect not to observe any missed detections or false alarms. All rows have Monte Carlo results are consistent with the predictions by the derived models.

for comparison against the model and a better-looking histogram. Fig. 5.22 provides a comparison histogram of the two scenarios and demonstrates that the model can predict the center and spread well with actual radio data.

### 5.5.6 Quick Application to Any GNSS

This dissertation has focused its example applications on SBAS. However, an important result of this chapter is that any GNSS can incorporate watermarking into its signal without utilizing additional data bandwidth and while maintaining backward compatibility. TESLA and the Combinatorial Watermarking construction enable a watermark without additional data bandwidth. The design methods from this section enable minimizing the needed watermark degradation, meaning that the watermark design could ensure that current, non-authenticating receivers barely notice a signal degradation. The most important design consideration is $\Theta$ since $\Theta$ will determine the authentication interval $W$, which will inversely affect $r$ in meeting a specific authentication security level.

I briefly discuss how to approach this for Galileo and encrypted signals in the following sections.

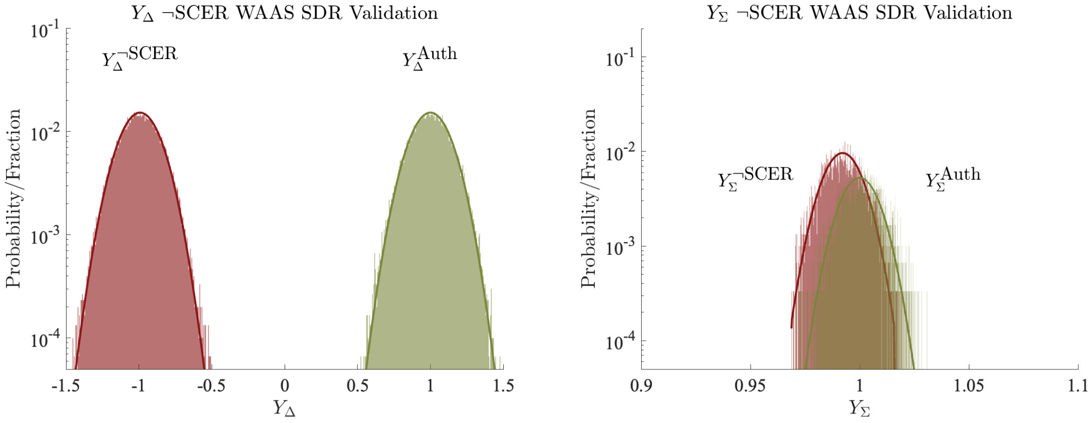


Figure 5.22: Non-SCER Distribution Validation For SBAS Watermark With SDR.

In this experiment, I compare real-world measurements of $Y_\Delta$ and $Y_\Sigma$ with the predicted distributions computed via convolution. The figure uses post-process WAAS PRN 131 C/A observations made with a 15.4 MHz SDR based on the mirrored experiment discussed in Section 5.5.5. Over a 1-hour time period, I measure 100 ms $Y_\Delta$ and $Y_\Sigma$. Each $Y$ included 100 1-ms coherent integrations via a converged tracking loop. Each millisecond contains its own watermark. In this scenario, the adversary and provider elected to invert $r = s = 4$ chips. From the mirror-problem $s$ election, the $Y_\Delta$ are separable and $Y_\Sigma$ distributions are close; however, the sum of them together are still less than the authentication threshold.

### Galileo

From Galileo's Open Service Navigation Message Authentication (OSNMA), $\Theta = 30$ seconds, which informs the maximum TESLA interval for which a watermark derives from a single hash point. The SBAS watermark for this chapter derives from the SBAS TESLA six-second interval and selects an $r$ given $W = 6000$. This leads to two initial watermark design starting points.

First, Galileo could adopt the watermark presented in this chapter without modification. This allows the receiver to observe random $W = 6000$ watermarks among the $W = 30000$ available per hash point. The advantage to the receiver is that it needs less memory storage since it would randomly store 6000 baseband ranging code measurements at a sampling rate of $F$. Second, applying the analysis from Section 5.5.2 to a 30-second Galileo interval, the watermark could be designed to include $r \approx 3$ with the same 32-bit security. While authenticating receivers will have to store and process all $W = 30000$ watermarks with sampling rate $F$, the non-authenticated receiver would observe a $2 * 3/1023 \approx 0.006 \approx$ -0.03 dB degradation in observed power.

### Encrypted Signals

With an encrypted ranging code, where the encryption serves to restrict access, the GNSS can simply watermark the encrypted ranging code. In the derivations of this section, $R$ will no longer be a constant ranging code (e.g., a Gold Code). Rather, $R$ will be a different encryption-derived ranging code per segment. Since the derivations are indifferent to the code underlying $R$, all of the methods still apply. And provided there is a TESLA protocol within the data component of the encrypted signal, GNSS can utilize the encrypted hash point distributions to watermark the encrypted ranging code.

# Chapter 6

# Conclusions

> GPS is a good ranging system but a terrible communication system.
>
> Jim Gillis

As discussed at the beginning of Chapter 1, there are primarily two audiences for this work: the Global Navigation Satellite System (GNSS) designer and the GNSS authentication researcher. The following two sections discuss the conclusions of this work for those two audience groups. After that, I finish with reviewing my contributions in context.

## 6.1 For the GNSS Designer

Chapter 2 provides a thorough treatment of the GNSS design considerations in selection a Timed Efficient Stream Loss-tolerant Authentication (TESLA) synchronization requirement $\Theta$. The GNSS designer must select $\Theta$ in consultation with the GNSS user base, understanding its effect on the time to authentication (TTA) and the accompanying onboard GNSS-independent clock (GIC) requirement. The tighter the $\Theta$, the faster the TTA achievable, and the greater the difficulty in procuring and maintaining a GIC. For ranging authentication, the tighter the $\Theta$, the greater the difficulty for a Security Code Estimation and Replay (SCER) adversary to break the ranging authentication security. The receiver must also hold its GIC time in confidence and ensure its GIC synchronization algorithms include the prescribed Network Time Security (NTS) modifications.

Chapter 3 leverages the temporal-geometric interpretation of cryptography to construct novel and easy-to-understand geometries useful to GNSS. Overall, the GNSS design should consolidate all authentication features into a single TESLA hash path. This consolidation is necessary for today's GNSS systems due to the severe data bandwidth limitation of modern GNSS signal designs. But for tomorrow's GNSS systems, this consolidation will enable lower authentication data bandwidth for optimally low TTA and quantum resistance. And for upcoming private, for-profit GNSS systems, this consolidation will allow bandwidth savings to be devoted to other profit-generating schemes.

With Chapters 3 and 4, the GNSS designer should have the tools needed to ensure that the product informs the cryptography rather than the cryptography informing the product. The GNSS cryptographer should not resist accommodating design requirements. Offhand, for a GNSS authentication scheme, 32 bits are required for each information authenticated with a CMT derived from the following transmitted hash point. Then, the transmission cadence, distributed in parallel among the constellation satellites, of the 128-bit hash points determines the TTA. For the cold-start receiver, the transmission cadence of the about 2000 TESLA maintenance bits, distributed in parallel among the constellation satellites, determines the time to first authenticated fix (TFAF).

Modern ranging codes evolved from dedicated research to advance the performance of the autocorrelation problem from Fig. 1.1. But, as users demand GNSS security, ranging codes will transition from modern ones (e.g., Gold, Weil) to cryptographic ones. Sadly, watermarks will erode those hard-fought code-based performance improvements. And cryptographic ranging codes purport to be indistinguishable from random codes, bounding their ranging performance. Just as skyscrapers grew taller by integrating their internal support systems into their external structure, GNSS authentication designs will cease separated form and function to shrink TTA by replacing modern ranging codes with cryptographic codes. The burden posed by a distinct ranging and cryptography mechanism will subside when cryptography serves both the authentication and the ranging mechanism. And when users eventually demand GNSS security with a fast TTA and as receivers become more integrated with network connections, GNSS signals will abandon their data components for cryptography-only signals like Fig. 3.18.

Chapter 5 demonstrates how signals can be augmented with a watermarks without burdening the signal's data bandwidth. Chapter 5 introduces Combinatorial Watermarking Functions and establishes that they provide a mathematical pathway for authentication security at a novel level of rigor. Their construction is flexible, and their security analysis is portable to different signals. From the procedure in Chapter 5, the GNSS designer can design a Combinatorial Watermark to meet requirements and prescribe the needed receiver signal processing. Moreover, the analysis provides a novel and concise understanding of the adversary's game to specify what the adversary can achieve.

## 6.2 For the GNSS Security Researcher

Chapter 2 provides the proofs needed for the authentication security checks regarding TESLA time synchronization in the GNSS broadcast-only context. Specifically, it specifies and proves the check needed on each authenticated information assuming a compliant GIC. It specifies and proves the checks needed to enforce a compliant GIC. Chapter 2 discusses a novel attack on GNSS TESLA receivers with broken clocks and discusses mitigations to address the vulnerability.

Chapter 3 discusses the rules establishing which temporal-geometric constructions are secure under GNSS TESLA. This enables constellation-wide TESLA safely to shrink TTA by parallelizing the hash point distribution. It enables combined watermarks to limit watermarking degradation for blended systems that accommodate multiple user groups. For Satellite-based Augmentation System (SBAS), Chapter 3 discusses how a TESLA design can accommodate SBAS alerts, provide core-GNSS-constellation navigation message authentication (NMA), and implement better error-correction codes for the CMTs.

Chapter 4 applies the maintenance design concepts to SBAS. It includes a method of delivering TESLA salt without data bandwidth by using Elliptic Curve Digital Signature Algorithm (ECDSA) signatures. The design is validated with the Matlab Algorithm Availability Simulation Tool (MAAST) showing acceptable performance with authentication added.

Chapter 5 introduces the Combinatorial Watermark and examines what makes a good watermarking function. The mathematical simplicity of Combinatorial Watermarks enables a mathematical pathway for a novel level of security rigor. The receiver statistics under spoofing conditions can be directly computed. Chapter 5 directly validates the distributions via Monte Carlo Methods and a novel experimental method leveraging a mirrored problem. For SBAS, Chapter 5 designs a watermark needing $r = 4$ chip inversions per ranging code. This amounts to a $-0.0341$ dB degradation.

### 6.2.1 Research Questions Remain

Chapter 2 presents an attack on the GNSS TESLA receiver resulting from when a receiver with a broken GIC reveals its broken state. While mitigations were provided, the question remains whether a synchronization protocol can provably obscure its current time state over multiple synchronizations.

Chapter 3 discusses how hash points can be distributed in parallel to decrease TTA, and provides a thorough treatment on random distribution strategies. While Chapter 3 briefly discusses the possibility of utilizing the constellation geometry to distribute hash points effectively, the question remains whether geometric strategies would provide a performance improvement guarantee worth implementing.

The formulations in Chapter 5 presume a secure estimate of the signal-to-noise ratio (SNR). To accommodate concerns on adversarial manipulation of these estimates, the designs in Chapter 5 assume unreasonably low noisy conditions during the watermark design phase. The question remains whether the security of SNR estimators can be shown and how that would affect Chapter 5's watermark design procedure.

For the Non-SCER adversary, Chapter 5 validates the authentication statistic distributions with present, real-world GNSS data. To do this without a real watermarked signal, Chapter 5 constructs a mirrored signal processing problem. This mirrored problem validation strategy has not yet been utilized to validate the authentication statistic distributions under the SCER adversary.

Finally, Chapter 5 discusses an SCER adversary that can exploit soft information when aggregating the chip estimations over an entire watermark. Via simulation, Chapter 5 shows that this adversary provides an advantage over the hard-decision security code estimation and replay (HDSCER). The question remains whether the SCER performance can be bound (given an adversary observed SNR), whether such a bound would be derived using error correction code soft information theory, and whether there are other soft-information SCER adversaries that beat the performance of the one presented in Chapter 5.

## 6.3 Contributions in Context

In Section 1.5, I provide the complete list of the claims and contributions of this thesis before their in-depth discussion in the following chapters. This section revisits them to remind the GNSS designer and research audience of their added value to next-generation GNSS.

Chapter 2 develops, aggregates, and proves the necessary synchronization protocols for receivers utilizing GNSS authentication (my first contribution). It compiles these protocols and warns of potential pitfalls in anticipated implementations for the GNSS designer. These contributions enable the convenient compilation provided in Chapter 2 with the hope that this topic will cease being a burdensome afterthought.

Chapter 3 designs efficient GNSS authentication constructions (my second contribution) that will not need to rely on de-minimus bit savings, inducing anxiety in cryptographers, to meet bandwidth constraints. Nor will GNSS designers of next-generation constellations need to make extraordinary adjustments to provide an authenticated product. Instead, they can rely on the techniques of Chapter 3 to minimize authentication data bandwidth or TTA. The product can dictate the cryptography rather than the reverse. Or the designer can utilize the crypto-first designs from Chapter 3 to push TTA performance.

Chapter 4 designs an SBAS authentication scheme with a TFAF of less than five minutes with present bandwidth limitations (my third contribution). The GNSS designer can use the SBAS example from Chapters 3 to 5 to design their authentication product.

From Chapter 5, GNSS designers and researchers can use a new framework to design and study ranging watermarks (my fourth and fifth contributions). The underlying math of the Combinatorial Watermark enables new performance optimizations (e.g., minimizing degradation), a pathway to understand a watermark's security and predict the effectiveness of an SCER adversary. Moreover, the constructions from Chapter 3 enable designers to combine watermarks to minimize degradation.

There are no more excuses: next-generation GNSS will have authentication.

# Bibliography

[1] Air Force Research Laboratory. *IS-AGT-100: Chips Message Robust Authentication (Chimera) Enhancement for the L1C Signal: Space Segment/User Segment Interface*. Tech. rep. Air Force Research Laboratory, Space Vehicles Director, 2019 (cit. on pp. xx, 36–38, 77, 175).

[2] Air Force Research Laboratory. *IS-AGT-101: Timed Efficient Stream Loss-tolerant Authentication (TESLA) Chips Message Robust Authentication (Chimera) Enhancement for the L1C Signal: Space Segment/User Segment Interface*. Tech. rep. Air Force Research Laboratory, Space Vehicles Director, 2024 (cit. on pp. xx, 37).

[3] Jason Anderson, Sherman Lo, and Todd Walter. "Addressing a Critical Vulnerability in Upcoming Broadcast-only TESLA-based GNSS-enabled Systems". In: *Proceedings of the 2023 International Technical Meeting of The Institute of Navigation*. 2023, pp. 277–285. DOI: `10.33012/2023.18623` (cit. on pp. 45, 47).

[4] Jason Anderson, Sherman Lo, and Todd Walter. "Authentication Security of Combinatorial Watermarking for GNSS Signal Authentication". In: *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*. 2023, pp. 495–509 (cit. on pp. 45, 169).

[5] Jason Anderson, Sherman Lo, and Todd Walter. “Authentication Security of Combinatorial Watermarking for GNSS Signal Authentication”. In: *NAVIGATION: Journal of the Institute of Navigation* 71.3 (2024). ISSN: 0028-1522. DOI: 10.33012/navi.655 (cit. on pp. 46, 169, 208).

[6] Jason Anderson, Sherman Lo, and Todd Walter. “Combinatorial Watermarking for GNSS Signal Authentication”. In: *Proceedings of the 2024 International Technical Meeting of The Institute of Navigation*. 2024, p. 314159. DOI: 10.33012/2024.19483 (cit. on pp. 46, 169, 173, 208, 222, 230).

[7] Jason Anderson, Sherman Lo, and Todd Walter. “Cryptographic Ranging Authentication with TESLA, Rapid Re-keying, and a PRF”. In: *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation*. 2022, pp. 43–55. DOI: 10.33012/2022.18226 (cit. on pp. 37, 45, 102).

[8] Jason Anderson, Sherman Lo, and Todd Walter. “Efficient and Secure Use of Cryptography for Watermarked Signal Authentication”. In: *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation*. 2022, pp. 68–82. DOI: 10.33012/2022.18228 (cit. on pp. 37, 45, 78, 102, 124, 125).

[9] Jason Anderson, Sherman Lo, and Todd Walter. “Revisiting Combinatorial Watermarking under SCER Adversarial Models”. In: *Proceedings of the ION 2024 Pacific PNT Meeting*. 2024, pp. 732–744. DOI: 10.33012/2024.19633 (cit. on pp. 46, 169).

[10] Jason Anderson, Sherman Lo, and Todd Walter. “Revisiting Combinatorial Watermarking under SCER Adversarial Models”. In: *NAVIGATION: Journal of the Institute of Navigation* (Accepted, pending revisions 2025) (cit. on pp. 46, 169).

[11] Jason Anderson, Sherman Lo, and Todd Walter. “Time Synchronization for TESLA-based GNSS-enabled Systems”. In: *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022)*. 2022, pp. 3408–3417. DOI: 10.33012/2022.18442 (cit. on pp. 45, 47).

[12] Jason Anderson, Sherman Lo, and Todd Walter. “Time Synchronization of TESLA-enabled GNSS Receivers”. In: *IEEE Transactions on Aerospace and Electronic Systems* (Accepted, pending revisions 2025) (cit. on pp. 46, 47).

[13] Jason Anderson et al. “Authentication of Satellite-Based Augmentation Systems with Over-the-Air Rekeying Schemes”. In: *NAVIGATION: Journal of the Institute of Navigation* 70.3 (2023). ISSN: 0028-1522. DOI: `10.33012/navi.595` (cit. on pp. 18, 34, 36, 46, 66, 68, 71, 97–99, 103, 104, 107, 132, 133, 137, 139–141, 143, 144, 148, 156, 162–164, 167, 231).

[14] Jason Anderson et al. “On SBAS Authentication with OTAR Schemes”. In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021)*. 2021, pp. 4288–4304 (cit. on pp. 45, 148).

[15] Jon M. Anderson et al. “Chips-message Robust Authentication (Chimera) for GPS Civilian Signals”. In: 2017, pp. 2388–2416. ISBN: 9781510853317. DOI: `10.33012/2017.15206` (cit. on pp. 38, 77).

[16] Robert Annessi et al. *Encryption is Futile: Delay Attacks on High-Precision Clock Synchronization.* 2018. DOI: `10.48550/ARXIV.1811.08569`. arXiv: `1811.08569` (cit. on pp. 2, 5, 7, 48, 57).

[17] Francesco Ardizzon et al. “It’s Galileo Time: Options for Crystal Oscillators in OSNMA-enabled Receivers”. In: *GPS World* (Jan. 2022) (cit. on p. 67).

[18] Markel Arizabaleta, Elias Gkougkas, and Thomas Pany. “A Feasibility Study and Risk Assessment of Security Code Estimation and Replay (SCER) Attacks”. In: *Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019)*. 2019, pp. 1039–1050. DOI: `10.33012/2019.17041` (cit. on p. 209).

[19] Georg T Becker et al. “Efficient Authentication Mechanisms for Navigation Systems-A Radio-navigation Case Study”. In: *Proceedings of the 22nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2009)*. 2009, pp. 901–912 (cit. on p. 34).

[20] Mihir Bellare and Tadayoshi Kohno. "A Theoretical Treatment of Related-key Attacks: RKA-PRPs, RKA-PRFs, and Applications". In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2003, pp. 491–506. DOI: `10.1007/3-540-39200-9_31` (cit. on pp. 12, 107).

[21] Ryad Benadjila et al. "SHA-3 Proposal: ECHO". In: *Submission to NIST (updated)* (2009), p. 113 (cit. on pp. 15, 19).

[22] Jon Louis Bentley. "A Sample of Brilliance". In: *Commun. ACM* 30.9 (1987), pp. 754–757. DOI: `10.1145/30401.315746` (cit. on pp. 181, 183).

[23] Joan Bernabeu et al. "A Collection of SDRs for Global Navigation Satellite Systems (GNSS)". In: *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation*. 2022, pp. 906–919. DOI: `10.33012/2022.18230` (cit. on pp. 194, 204).

[24] John W. Betz. "Fundamentals of Satellite-Based Navigation and Timing". In: *Position, Navigation, and Timing Technologies in the 21st Century*. John Wiley & Sons, Ltd, 2020. Chap. 2, pp. 43–64. ISBN: 9781119458449. DOI: `10.1002/9781119458449.ch2` (cit. on pp. 3–5).

[25] Alex Biryukov and Dmitry Khovratovich. "Related-key Cryptanalysis of the Full AES-192 and AES-256". In: *Advances in Cryptology–ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings 15*. Springer. 2009, pp. 1–18. DOI: `10.1007/978-3-642-10366-7_1` (cit. on pp. 12, 107).

[26] Christian Blatter. *What's the proof of correctness for Robert Floyd's algorithm for selecting a single, random combination of values?* Mathematics Stack Exchange. URL: `https://math.stackexchange.com/q/178723` (cit. on p. 183).

[27] M. Blaum et al. "EVENODD: An Efficient Scheme for Tolerating Double Disk Failures in RAID Architectures". In: *IEEE Transactions on Computers* 44.2 (1995), pp. 192–202. DOI: `10.1109/12.364531` (cit. on p. 142).

[28] Dan Boneh and Victor Shoup. “A Graduate Course in Applied Cryptography”. In: (2023). URL: `https://toc.cryptobook.us` (cit. on pp. 11, 26).

[29] C4ADS. *Above Us Only Stars. Exposing GPS Spoofing in Russia and Syria.* 2019 (cit. on p. 8).

[30] Simón Cancela, J. David Calle, and Ignacio Fernández-Hernández. “CPU Consumption Analysis of TESLA-based Navigation Message Authentication”. In: *2019 European Navigation Conference (ENC)* (2019), pp. 1–6. DOI: `10.1109/EURONAV.2019.8714171` (cit. on p. 161).

[31] Gianluca Caparra and James T Curran. “On the Achievable Equivalent Security of GNSS Ranging Code Encryption”. In: *2018 IEEE/ION Position, Location and Navigation Symposium (PLANS).* IEEE. 2018, pp. 956–966. DOI: `10.1109/PLANS.2018.8373474` (cit. on p. 171).

[32] Gianluca Caparra et al. “Evaluating the Security of One-way Key Chains in TESLA-based GNSS Navigation Message Authentication Schemes”. In: *2016 International Conference on Localization and GNSS (ICL-GNSS)* (2016), pp. 1–6. DOI: `10.1109/ICL-GNSS.2016.7533685` (cit. on pp. 14, 20, 140, 160).

[33] S Damy, L Cucchi, and M Paonni. “Performance Assessment of Galileo OSNMA Data Retrieval Strategies”. In: *Proceedings of the NAVITEC 2022 conference.* 2022, pp. 5–7 (cit. on p. 162).

[34] Sophie Damy, Luca Cucchi, and Matteo Paonni. “Impact of OSNMA Configurations, Operations and User’s Strategies on Receiver Performances”. In: *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022).* 2022, pp. 820–827. DOI: `10.33012/2022.18302` (cit. on p. 162).

[35] Quynh Dang. *The Keyed-Hash Message Authentication Code (HMAC).* en. July 2008. DOI: `10.6028/NIST.FIPS.198-1` (cit. on p. 19).

[36] Frank van Diggelen. “Assisted GNSS”. In: *Position, Navigation, and Timing Technologies in the 21st Century.* John Wiley & Sons, Ltd, 2020. Chap. 17,

pp. 419–444. ISBN: 9781119458449. DOI: `10.1002/9781119458449.ch17` (cit. on pp. 137, 138).

[37] D. Dolev and A. Yao. “On the Security of Public Key Protocols”. In: *IEEE Transactions on Information Theory* 29.2 (1983), pp. 198–208. DOI: `10.1109/TIT.1983.1056650` (cit. on p. 60).

[38] EUSPA. “Galileo Open Service Navigation Message Authentication (OSNMA) Receiver Guidelines”. In: vol. 1. Jan. 2024 (cit. on pp. 140, 141, 147).

[39] EUSPA. “Galileo Open Service Navigation Message Authentication (OSNMA) Signal-In-Space Interface Control Document (SIS OCD)”. In: vol. 1. Oct. 2023 (cit. on pp. 35, 66, 119, 151, 152).

[40] Emanuela Falletti, Marco Pini, and Letizia Lo Presti. “Low Complexity Carrier-to-Noise Ratio Estimators for GNSS Digital Receivers”. In: *IEEE Transactions on Aerospace and Electronic Systems* 47.1 (2011), pp. 420–437. DOI: `10.1109/TAES.2011.5705684` (cit. on p. 194).

[41] Ignacio Fernandez-Hernandez. “Snapshot and Authentication Techniques for Satellite Navigation”. PhD thesis. June 2015 (cit. on pp. 18, 32, 105, 162).

[42] IGNACIO Fernandez-Hernandez et al. “Galileo Authentication and High accuracy: Getting to the Truth”. In: *Inside GNSS* (2023) (cit. on p. 35).

[43] Ignacio Fernandez-Hernandez et al. “Fountain Codes for GNSS”. In: *Proceedings of the 30th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2017)* (Sept. 2017). DOI: `https://doi.org/10.33012/2017.15368` (cit. on p. 162).

[44] Ignacio Fernandez-Hernandez et al. “Increasing International Civil Aviation Resilience: A Proposal for Nomenclature, Categorization and Treatment of New Interference Threats”. In: *Proceedings of the 2019 International Technical Meeting of The Institute of Navigation* (Jan. 2019). DOI: `10.33012/2019.16699` (cit. on p. 1).

[45] Ignacio Fernandez-Hernandez et al. “Independent Time Synchronization for Resilient GNSS Receivers”. In: *Proceedings of the 2020 International Technical Meeting of The Institute of Navigation*. 2020, pp. 964–978. DOI: `10.33012/2020.17190` (cit. on p. 28).

[46] Ignacio Fernandez-Hernandez et al. “Message Authentication Candidates for the SBAS Dual Frequency Multi-Constellation Standard”. In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021)*. 2021, pp. 443–452. DOI: `10.33012/2021.17892` (cit. on p. 44).

[47] Ignacio Fernandez-Hernandez et al. “SBAS Message Authentication: A review of Protocols, Figures of Merit and Standardization Plans”. In: *Proceedings of the 2021 International Technical Meeting of The Institute of Navigation* (Jan. 2021). DOI: `10.33012/2021.17829` (cit. on p. 138).

[48] Ignacio Fernández-Hernández, Tomer Ashur, and Vincent Rijmen. “Analysis and Recommendations for MAC and Key Lengths in Delayed Disclosure GNSS Authentication Protocols”. In: *IEEE Transactions on Aerospace and Electronic Systems* 57.3 (2021), pp. 1827–1839. DOI: `10.1109/TAES.2021.3053129` (cit. on pp. 17, 18, 41).

[49] Ignacio Fernández-Hernández et al. “A Navigation Message Authentication Proposal for the Galileo Open Service”. In: *NAVIGATION* 63.1 (2016), pp. 85–102. DOI: `10.1002/navi.125` (cit. on pp. 18, 35).

[50] Ignacio Fernández-Hernández et al. “SBAS Message Authentication: A Review of Protocols, Figures of Merit and Standardization Plans”. In: *Proceedings of the 2021 International Technical Meeting of The Institute of Navigation*. 2021, pp. 111–124. DOI: `10.33012/2021.17829` (cit. on pp. 35, 44).

[51] Daniel Fox Franke et al. *Network Time Security for the Network Time Protocol*. RFC 8915. Sept. 2020. DOI: `10.17487/RFC8915` (cit. on p. 56).

[52] Aleix Galan et al. *Improving Galileo OSNMA Time To First Authenticated Fix.* 2024. arXiv: `2403.14739 [cs.CR]`. URL: `https://arxiv.org/abs/2403.14739` (cit. on p. 32).

[53] Martin Götzelmann et al. “Galileo Open Service Navigation Message Authentication: Preparation Phase and Drivers for Future Service Provision”. In: *NAVIGATION: Journal of the Institute of Navigation* 70.3 (2023). ISSN: 0028-1522. DOI: `10.33012/navi.572` (cit. on pp. 18, 35).

[54] Committee on Graduate Studies. *Stanford Graduate Academic Policies and Procedures 4.8.1: Doctoral Degrees, Dissertations & Dissertation Reading Committees: Policy.* `https://gap.stanford.edu/handbooks/gap-handbook/chapter-4/subchapter-8/page-4-8-1`. Mar. 2022 (cit. on p. 41).

[55] Lov K. Grover. “A Fast Quantum Mechanical Algorithm for Database Search”. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing.* STOC ’96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219. ISBN: 0897917855. DOI: `10.1145/237814.237866` (cit. on p. 30).

[56] Joanna Hinks et al. “Signal and Data Authentication Experiments on NTS-3”. In: Sept. 2021, pp. 3621–3641. DOI: `10.33012/2021.17964` (cit. on p. 38).

[57] Todd Humphreys. “Detection Strategy for Cryptographic GNSS Anti-Spoofing”. In: *Aerospace and Electronic Systems, IEEE Transactions on* 49 (Apr. 2013), pp. 1073–1090. DOI: `10.1109/TAES.2013.6494400` (cit. on pp. 210, 211).

[58] Todd Humphreys. “Limitations of Signal-side GNSS Signal Authentication”. In: Sept. 2014, pp. 1351–1363 (cit. on p. 5).

[59] “IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems”. In: *IEEE Std 1588-2019 (Revision ofIEEE Std 1588-2008)* (2020), pp. 1–499. DOI: `10.1109/IEEESTD.2020.9120376` (cit. on p. 56).

[60] Andrew J. Kerns, Kyle D. Wesson, and Todd E. Humphreys. "A Blueprint for Civil GPS Navigation Message Authentication". In: *2014 IEEE/ION Position, Location and Navigation Symposium - PLANS 2014*. 2014, pp. 262–269. DOI: 10.1109/PLANS.2014.6851385 (cit. on p. 34).

[61] Anargyros Kriezis et al. "Identifying Low Cost GNSS Monitor Metrics for Robust RFI Detection". In: *Proceedings of the 2024 International Technical Meeting of The Institute of Navigation*. 2024, pp. 426–440. DOI: 10.33012/2024.19542 (cit. on p. 8).

[62] S. Lo et al. "Signal Authentication: A Secure Civil GNSS for Today". In: *Inside GNSS* 4 (Sept. 2009), pp. 30–39. URL: https://web.stanford.edu/group/scpnt/gpslab/pubs/papers/Lo_InsideGPS_SepOct2009.pdf (cit. on p. 40).

[63] Sherman C. Lo and Per K. Enge. "Authenticating Aviation Augmentation System Broadcasts". In: *IEEE/ION Position, Location and Navigation Symposium*. 2010, pp. 708–717. DOI: 10.1109/PLANS.2010.5507223 (cit. on p. 102).

[64] Jim Martin et al. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905. June 2010. DOI: 10.17487/RFC5905 (cit. on p. 56).

[65] Ralph C Merkle. "A Digital Signature Based on a Conventional Encryption Function". In: *Conference on the theory and application of cryptographic techniques*. Springer. 1987, pp. 369–378. DOI: 10.1007/3-540-48184-2_32 (cit. on p. 151).

[66] Tara Mina, Alan Yang, and Grace Gao. "Designing Long GPS Memory Codes Using the Cross Entropy Method". In: *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*. 2023, pp. 1328–1340. DOI: 10.33012/2023.19260 (cit. on p. 113).

[67] Luciano Musumeci et al. "OSNMA User Performance Assessment at ESA/ESTEC–System Qualifications Tools and Methodologies". In: *Proceedings of the 36th International Technical Meeting of the Satellite Division of*

*The Institute of Navigation (ION GNSS+ 2023)*. 2023, pp. 538–556. DOI: 10.33012/2023.19224 (cit. on pp. 32, 35).

[68] Lakshay Narula and Todd E. Humphreys. “Requirements for Secure Clock Synchronization”. In: *IEEE Journal of Selected Topics in Signal Processing* 12.4 (2018), pp. 749–762. DOI: 10.1109/JSTSP.2018.2835772 (cit. on pp. 5, 50, 59, 61).

[69] National Institute of Standards and Technology (NIST). *Post-Quantum Cryptography Standardization*. Tech. rep. Accessed: 2024-08-06. National Institute of Standards and Technology, 2017. URL: https://csrc.nist.gov/projects/post-quantum-cryptography (cit. on pp. 30, 154).

[70] Andrew Neish. “Establishing Trust through Authentication in Satellite Based Augmentation Systems”. PhD thesis. Stanford, CA, 2020 (cit. on pp. 32, 35, 36, 41, 103, 105, 138–141, 163, 164).

[71] Andrew Neish, Todd Walter, and Per Enge. “Parameter Selection for the TESLA Keychain”. In: *Proceedings of the 31st international technical meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2018)*. 2018, pp. 2155–2171. DOI: 10.33012/2018.15852 (cit. on p. 20).

[72] Andrew Neish, Todd Walter, and Per Enge. “Quantum-resistant Authentication Algorithms for Satellite-based Augmentation Systems”. In: *Navigation* 66.1 (2019), pp. 199–209. DOI: 10.33012/2018.15538 (cit. on p. 35).

[73] Andrew Neish, Todd Walter, and J. David Powell. “Design and Analysis of a Public Key Infrastructure for SBAS Data Authentication”. In: *Journal of the Institute of Navigation* 66.4 (2019), pp. 831–844. DOI: 10.1002/navi.338 (cit. on p. 138).

[74] Andrew N. Novick and Michael A. Lombardi. “Practical Limitations of NTP Time Transfer”. In: *2015 Joint Conference of the IEEE International Frequency Control Symposium & the European Frequency and Time Forum*. 2015, pp. 570–574. DOI: 10.1109/FCS.2015.7138909 (cit. on p. 95).

[75] C O'Driscoll et al. *The Attack Agnostic Defence: a Spoofing Detection Metric for Secure Spreading Sequences*. 2022 (cit. on pp. 227, 228).

[76] Brady O'Hanlon et al. "SBAS Signal Authentication". In: *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022)*. 2022, pp. 3369–3377. DOI: `10.33012/2022.18443` (cit. on pp. 44, 172).

[77] Philippe Oechslin. "Making a Faster Cryptanalytic Time-memory Trade-off". In: *Advances in Cryptology-CRYPTO 2003: 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003. Proceedings 23*. Springer. 2003, pp. 617–630. DOI: `10.1007/978-3-540-45146-4_36` (cit. on pp. xxi, 14).

[78] ICAO (International Civil Aviation Organziation). *Aeronautical Information Regulation and Control (AIRAC)*. Accessed: 2024-08-26. 2021. URL: `https://www.icao.int/airnavigation/information-management/Pages/AIRAC.aspx` (cit. on pp. 68, 150).

[79] ICAO (International Civil Aviation Organziation). *Standards and Recommended Practices (SARPS) Annex 10 to the Convention on International Civil Aviation*. Tech. rep. July 2023 (cit. on pp. 132, 139, 143).

[80] Adrian Perrig et al. "Timed Efficient Stream Loss-tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction". In: *Request For Comments (RFC)* 4082 (2005) (cit. on pp. 20, 27, 59).

[81] Thomas Peyrin, Yu Sasaki, and Lei Wang. "Generic Related-Key Attacks for HMAC". In: *Advances in Cryptology – ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 580–597. ISBN: 978-3-642-34961-4. DOI: `10.1007/978-3-642-34961-4_35` (cit. on pp. 12, 107).

[82] Christoph Ponikwar et al. *Beyond the Dolev-Yao Model: Realistic Application-Specific Attacker Models for Applications Using Vehicular Communication*. 2016. DOI: `10.48550/ARXIV.1607.08277` (cit. on p. 60).

[83] Mark L. Psiaki and Todd E. Humphreys. "GNSS Spoofing and Detection". In: *Proceedings of the IEEE* 104.6 (2016), pp. 1258–1270. DOI: 10.1109/JPROC.2016.2526658 (cit. on pp. 1, 5).

[84] Mark L. Psiaki et al. "GPS Spoofing Detection via Dual-Receiver Correlation of Military Signals". In: *IEEE Transactions on Aerospace and Electronic Systems* 49.4 (2013), pp. 2250–2267. DOI: 10.1109/TAES.2013.6621814 (cit. on p. 40).

[85] Tyler G.R. Reid et al. "Satellite Navigation for the Age of Autonomy". In: *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. 2020, pp. 342–352. DOI: 10.1109/PLANS46316.2020.9109938 (cit. on p. 1).

[86] Fabian Rothmaier et al. "GNSS Spoofing Detection through Spatial Processing". In: *NAVIGATION* 68.2 (2021), pp. 243–258. DOI: 10.1002/navi.420 (cit. on pp. 1, 8).

[87] RTCA, Inc. *Minimum Operational Performance Standards for Global Positioning System/Satellite-Based Augmentation System Airborne Equipment*. Tech. rep. DO-229F. Includes specifications for SBAS-capable GPS receivers. RTCA, Inc., June 2020. URL: https://www.rtca.org/publications/ (cit. on p. 132).

[88] JOSEPH J. RUSHANAN. "The Spreading and Overlay Codes for the L1C Signal". In: *NAVIGATION* 54.1 (2007), pp. 43–51. DOI: 10.1002/j.2161-4296.2007.tb00394.x (cit. on p. 175).

[89] Joseph J. Rushanan. "Weil Sequences: A Family of Binary Sequences with Good Correlation Properties". In: *2006 IEEE International Symposium on Information Theory*. 2006, pp. 1648–1652. DOI: 10.1109/ISIT.2006.261556 (cit. on p. 113).

[90] Claus-Peter Schnorr. "Efficient Signature Generation by Smart Cards". In: *Journal of cryptology* 4 (1991), pp. 161–174. DOI: 10.1007/BF00196725 (cit. on pp. 17, 19).

[91] L. Scott. “Anti-Spoofing & Authenticated Signal Architectures for Civil Navigation Systems”. In: *Proceedings of the 16th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS/GNSS 2003)*. 2003, pp. 1543–1552 (cit. on pp. 1, 169).

[92] Jeff A Sherman and Judah Levine. “Usage Analysis of the NIST Internet Time Service”. In: *Journal of Research of the National Institute of Standards and Technology* 121 (2016), p. 33. DOI: `10.6028/jres.121.003` (cit. on p. 97).

[93] P.W. Shor. “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134. DOI: `10.1109/SFCS.1994.365700` (cit. on p. 30).

[94] F. Soualle et al. “Spreading Code Selection Criteria for the future GNSS Galileo”. In: *Proc. of the European Navigation Conf. GNSS* (2005), pp. 1–10 (cit. on p. 113).

[95] National Institute of Standards and Technology. *Advanced Encryption Standard (AES)*. Tech. rep. 2023. DOI: `10.6028/NIST.FIPS.197-upd1` (cit. on p. 19).

[96] National Institute of Standards and Technology. *Digital Signature Standard (DSS)*. Tech. rep. 2023. DOI: `10.6028/NIST.FIPS.186-5` (cit. on p. 19).

[97] National Institute of Standards and Technology. *Implementation Guidance for FIPS 140-3 and the Cryptographic Module Validation Program*. Tech. rep. 2024 (cit. on p. 15).

[98] National Institute of Standards and Technology. *Keyed-Hash Message Authentication Code (HMAC): Specification of HMAC and Recommendations for Message Authentication*. Tech. rep. NIST SP 800-224 Inidial Public Draft. Washington, D.C.: U.S. Department of Commerce, 2020. DOI: `10.6028/NIST.SP.800-224.ipd` (cit. on p. 17).

[99] National Institute of Standards and Technology. *Keyed-Hash Message Authentication Code (HMAC): Specification of HMAC and Recommendations for Message Authentication*. Tech. rep. 2024. DOI: `10.6028/NIST.SP.800-224.ipd` (cit. on p. 19).

[100] National Institute of Standards and Technology. *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*. Tech. rep. 2024. DOI: `10.6028/NIST.SP.800-38D` (cit. on pp. 19, 26).

[101] National Institute of Standards and Technology. *Recommendation for Key Derivation Using Pseudorandom Functions*. Tech. rep. 2024. DOI: `10.6028/NIST.SP.800-108r1-upd1` (cit. on p. 19).

[102] National Institute of Standards and Technology. *Recommendation for Key Management: Part 1 – General*. Tech. rep. NIST SP 800 Part 1 Rev. 5. Washington, D.C.: U.S. Department of Commerce, 2020. DOI: `10.6028/NIST.SP.800-57pt1r5` (cit. on p. 157).

[103] National Institute of Standards and Technology. *Secure Hash Standard (SHS)*. Tech. rep. 2015. DOI: `10.6028/NIST.FIPS.180-4` (cit. on pp. 15, 19).

[104] National Institute of Standards and Technology. *SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash, and ParallelHash*. Tech. rep. 2023. DOI: `10.6028/NIST.SP.800-185` (cit. on p. 19).

[105] National Institute of Standards and Technology. *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. Tech. rep. 2023. DOI: `10.6028/NIST.FIPS.202` (cit. on pp. 15, 19).

[106] Marc Stevens et al. "The First Collision for Full SHA-1". In: *Advances in Cryptology–CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20–24, 2017, Proceedings, Part I 37*. Springer. 2017, pp. 570–596. DOI: `10.1007/978-3-319-63688-7_19` (cit. on pp. 15, 19).

[107] Rafael Terris-Gallego et al. "Guidelines for Galileo Assisted Commercial Authentication Service Implementation". In: *2022 International Conference on Localization and GNSS (ICL-GNSS)*. 2022, pp. 01–07. DOI: `10.1109/ICL-GNSS54081.2022.9797027` (cit. on p. 40).

[108] Todd Walter, Jason Anderson, and Sherman Lo. "Implementation of Data Authentication on SBAS". In: *Proceedings of the ION 2024 Pacific PNT Meeting*. 2024, pp. 709–721. DOI: `10.33012/2024.19631` (cit. on pp. 45, 104, 105, 139, 141, 142).

[109] Todd Walter, Jason Anderson, and Sherman Lo. "SBAS Message Schemes to Support Inline Message Authentication". In: *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021)*. 2021, pp. 474–484. DOI: `10.33012/2021.17908` (cit. on p. 44).

[110] Wikipedia. *Elliptic Curve Digital Signature Algorithm — Wikipedia, The Free Encyclopedia*. [Online; accessed 21-August-2024]. 2024. URL: `https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm` (cit. on p. 159).