Università degli Studi di Padova

Department of Information Engineering

Ph.D. Course: Information Engineering
Curriculum: Information Science and Technology
Series: XXXII

Security in Global Navigation Satellite Systems: authentication, integrity protection and access control

Ph.D. Student:
Silvia Ceccato

Supervisor:
Professor Nicola Laurenti

Coordinator:
Professor Andrea Neviani

Academic Year: 2018–2019

Abstract

In the last decades the use of Global Navigation Satellite System (GNSS) positioning has spread through numerous commercial applications and permeated our daily life. Matching this growing interest for precise positioning worldwide, new systems have been designed and deployed, such as Galileo, the European GNSS. As these systems are relied upon in an ever growing number of safety critical applications, it is vital to devise protections and countermeasures against any threats that may target GNSS modules to harm underlying service. The potential economical advantage that derives from disrupting or manipulating the service is indeed an incentive for malicious users to devise smarter and more sophisticated threats.

This thesis tackles the evolution of attacks and corresponding security measures in GNSS, investigating state-of-the-art approaches from both the attacker's and the system's point of view. The work focuses on various security targets, such as authentication, integrity protection and access control, exploring threats and solutions at both signal and data level. Securing GNSS from malicious entities indeed requires protecting all of its components: the navigation message, the signal-in-space and the computed Position, Velocity and Time (PVT). All three domains are investigated with the aim of assessing the vulnerability of the system to state-of-the-art threats and providing guidelines for the addition of future security features.

Contents

List of Figures

Acronyms

CA	Certificate Authority	51
A-GNSS	Assisted GNSS	6
BMBE	broadcast encryption scheme based on bilinear maps	74
C/N_0	carrier-to-noise-floor ratio	vii
CRC	cyclic redundancy check	23
CDF	Cumulative Distribution Function	63
CDMA	code-division multiple access	3
DET	detection error tradeoff	viii
DoS	Denial of Service	9
DOP	dilution of precision	22
DSA	Digital Signature Algorithm	15
EKF	Extended Kalman Filter	135
EC	European Commission	18
ESA	European Space Agency	7
FEA	Forward Estimation Attack	9
FEC	forward-error correction	17
GNSS	Global Navigation Satellite System	iii
GPS	Global Positioning System	viii
ICD	Iterface Control Document	111
i.i.d.	independent and identically distributed	66
IMU	Inertial Measurement Unit	6
IOD	Issue Of Data	56
KM	Key Management	4
KF	Kalman Filter	6
KMM	Key Management Message	59
LOS	line-of-sight	37

Chapter 1

Introduction

GNSSs provide accurate, continuous, worldwide, three-dimensional position, velocity and time information. While the GPS has been part of many people's daily experience worldwide for decades, in recent years, there has been an ever growing dependence on GNSS for applications in sectors ranging from telecommunications, energy transmission and distribution, to financial services and transportation.

GNSS is however not exempt from vulnerabilities: it is sensitive to interference, either incidental or intentional, that may endanger the availability of service [1]. Any information needed for the generation of civilian use GNSS signals is public. This constitutes a valuable advantage in that signals can be easily reproduced for design and testing purposes, allowing to simulate real world scenarios. On the other hand, if no authentication mechanisms are implemented, the signal can be as easily falsified by malicious entities. As the demand for reliable navigation for commercial applications has increased, so too have risks of intentional interference or spoofing of GNSS signals from adversaries with the intent of causing damage or obtaining illegitimate advantage.

In fact, two kinds of threats are particularly relevant to the GNSS context: *jamming*, carried out at the physical layer and aiming at denying the service to receivers in a given area, and *spoofing*, that aims to induce the receiver into computing a false position, velocity or time, by forging or illegitimately modifying the GNSS signal either at the data/message level or at the code/signal level.

The growing interest in GNSS has brought the European Union to develop its own system, Galileo, which recently became operational. To answer the arising concerns on GNSS security, the European Commission has recently announced that Galileo will offer Navigation Message Authentication (NMA) as a protection against falsified signals, such as meaconing and spoofing attacks. While the primary purpose of NMA is to provide navigation message assurance, it has been suggested in the literature that it could also play a role in providing some form of range assurance through the exploitation of symbol-level entropy, i.e., the unpredictability of symbols in the navigation message. This claim is addressed in Chapter 3, where the effectiveness of data-level techniques for the mitigation of spoofing attacks is assessed through theoretical and experimental analyses.

Most of the GNSS signals are based on code-division multiple access (CDMA): the transmitted signal is obtained by multiplication with a spreading sequence, called Pseudo-Random Noise (PRN) code, that spreads the signal power over a larger bandwidth.

The GNSS signal is received below the thermal noise and a correlation with the known spreading sequence is performed in order to recover the signal. However, the low received power makes GNSS vulnerable to RF interference, either intentional (i.e., jamming) or unintentional. The most trivial approach for jamming attacks can be seen as brute-force and is based on transmitting a high power sinusoidal wave. This signal disrupts the service over a certain area rather than selectively targeting a particular device or signal. Several interference mitigation techniques are based on the detection and eviction of narrow-band interference signals in the frequency domain, where they have a sparse representation, as in [2]. The growing interest in commercial GNSS applications, however, has led towards the investigation of more subtle interference strategies. The rationale behind this research is leveraging the structure of the GNSS signal to disrupt the positioning service more efficiently than with traditional jamming approaches. In Chapter 4 a novel jamming technique is examined that targets each ranging signal individually, aiming at disrupting the Position, Navigation and Timing (PNT) capability for receivers located in a specific position, by directly attacking the correlation process.

The computation of a PVT solution requires two types of information: the positions of at least four SVs derived from the navigation message and the distances between the receiver and each SV, that must be estimated from the received ranging signal. Therefore, a malicious entity that aims at inducing a false PVT solution onto a victim receiver can either leverage a forged (or altered) navigation message, a spoofed ranging signal, or both. Thus the authenticity of GNSS needs to be ensured both at the data and signal layers, in the form of cryptographic integrity and signal authenticity.

At both layers, the protection of GNSS is supported at the system side and often relies on some cryptographic scheme (e.g., NMA) or function (e.g., Spreading Code Encryption (SCE)) for signing or watermarking sensible information. The security of any such scheme relies on the keys that are shared or distributed among the legitimate parties. The creation, distribution, renewal and revocation of cryptographic keys are regulated by what is called the Key Management (KM) infrastructure, which represents an ancillary service to the cryptographic mechanisms. Chapter 5 introduces the concept of key management and is divided in two sections, each dealing with one particular security target: authentication and access control.

Current proposals for NMA [3–11] have investigated the adaptation of existing authentication and integrity protection mechanisms to GNSS. Some of these works [7–10] have also mentioned the need of a dedicated key management protocol for NMA. However, the design of a dedicated key management architecture still remains an open topic. In Sec. 5.1 the design of a key management infrastructure for GNSS authentication schemes is tackled by drawing primitives and key distribution approaches from well known communications systems. The proposed key management scheme is based on a layered structure, where higher layer keys, more secure and longer valid, protect the integrity of messages for the management of lower layer keys. The lowest layer keys are used for the NMA mechanism and are frequently changed. Moreover, in order to save bandwidth they are stored into the receiver in encrypted form, together with their certificate, to be decrypted and retrieved as they come into use. The proposed architecture is the result of a trade off between communication overhead and storage requirements at the user's side, with the aim of offering a solution that is tailored to the system's constraints.

Authentication is not the only security target in GNSS. The European Commission has envisioned a *Commercial Service* signal for Galileo, that will offer added value services with respect to the Open Service (OS). While the service concept has not been finalized yet, one of the possibilities is that it may be provisioned on demand and its access limited to subscribed users. Thus, an architecture must be devised in order to protect the revenue return through *access control*, a cryptographic function that allows to regulate the access to resources through encryption. While key management for authentication can be devised as an additional layer independently of the authentication scheme, the relationship between the access control scheme and its key management layer is tighter. Access control schemes indeed are in general designed together with a built in key distribution and update architecture, [12–16]. Sec. 5.3 tackles the design of a key management scheme for access control to broadcast services. The scheme is devised for a broader category of communication systems that share features such as the low data rate and the multicast nature of the service provision.

GNSS are critical infrastructures operating on a continental scale. Their operations depend on the delivery of different types of information and data from the ground segment to the SV, orbiting at about 20000km altitude. These communications are currently carried out through radio links. However some works in literature are exploring the path of optical communications: the first experimental feasibility study of an optical link between a ground station and a GNSS terminal has been presented in [17]. The objective of this study is to prove the feasibility of the implementation of ground-satellite and inter-satellite Quantum Key Distribution (QKD) in GNSS. QKD enables unconditionally secure generation of cryptographic keys and opens the road towards novel protocols for authentication, integrity protection and access control to the exchanged signals. Even though classic cryptographic schemes (e.g.: RSA, Diffie-Hellmann) are well established, standardized and commonly used in commercial applications, their security is founded on the assumption that the attacker's computational power is limited. In the near future this assumption may be mined by breakthroughs in the research on quantum computers. Therefore it is essential to devise an alternative solution for security critical infrastructures such as the Galileo constellation [18]. The proved feasibility of inter-satellite quantum links enables an inter satellite QKD function, allowing SVs to autonomously exchange criptographic material in an unconditionally secure fashion. Since the inter-satellite links are feasible only between certain satellite pairs, a multi-hop key distribution protocol must be devised, exploiting the secure bits exchanged through quantum links. The security targets of this key agreement mechanism are discussed in Chapter 6, where an algorithm is devised for the routing function of key distribution in the satellite network. The devised QKD algorithm is extended by integrating Shamir's polynomial threshold secret sharing [19] in order to enhance the protocol's resilience to satellite eavesdropping or hacking.

Despite the criticality of GNSS as a global system, the applications that use it are numerous and diverse, spanning from the highly sensitive military sector to low impact aspects of people's everyday life. This diversity must be taken into account when devising anti-spoofing solutions at the receiver's side, as security requirements depend on the specific application. The cryptographic mechanisms reviewed so far are implemented at the system's side and, despite their value as an anti-spoofing enabler, they may not be the most efficient solution for low-grade applications. While autonomous receivers solely rely on GNSS to calculate their position and time, mobile handsets benefit from other options. Other sources, such as signals of opportunity (WiFi, cellular networks, Bluetooth) or the on board sensors (accelerometer, digital compass, clock, etc.) provide useful redundancy that is currently exploited to improve the efficiency and availability of positioning services (e.g., indoor positioning, dead reckoning in tunnels, etc.). Assisted GNSS (A-GNSS) has contributed to the widespread of GNSS positioning in handsets, allowing mobile devices to retrieve system information (ephemeris data, frequency and code delay estimates, etc.) from an aiding channel. These resources have greatly improved the positioning performance of GNSS based applications, allowing mobile devices to provide a PVT solution even in the most challenging conditions. Since GNSS spoofing is now an emerging issue, there is no reason why these beneficial sources should not be used as an ally for security purposes as well. In Chapter 7 the resilience of mobile devices to GNSS spoofing is investigated through an extensive experimental campaign. Some simple consistency checks are highlighted that would increase the user awareness, minimizing the impact on the user segment without requiring modifications to the ground or space segment.

Commercial based application that rely on navigation have high interest in implementing robust and redundant spoofing detection mechanisms, but often have to cope with limited budget. Since high integrity tactical grade GNSS receiver are in general not available in these scenarios, integrating all on-board sources of positioning redundancy is the most efficient way to devise a robust yet affordable spoofing detection architecture. Chapter 8 will tackle the challenges of this approach tailored to automotive applications. The focus is on the integration of data from the Inertial Measurement Unit (IMU) in a Kalman Filter (KF) framework in order to detect any inconsistency between the actual body movement and the trajectory derived from GNSS.

GNSS is not the only system that provides positioning information to commercial applications. Cellular networks from the third to the upcoming fifth generation adopt several standardized protocols for estimating the user's position. Fourth generation networks even employ some dedicated ranging sequences that allow to perform trilateration in a similar fashion as GNSS. As the interest on location awareness for a wide variety applications has grown, so will the incentive to disrupt or manipulate the position information for gaining any sort of economical advantage (e.g., position based road tolling or parking management applications). Chapter 9 focuses on the problem of authenticating users' position to network authorities. A preliminary investigation is carried out to assess the feasibility of an authentication protocol based on two dimensional distance bounding. The protocol exploits the rich geometry derived from network densification and assumes the use of an independent signal component.

The focus of this Ph.D. was not only on research, but involved the development of a GNSS software simulator and a GNSS software receiver. The software package, introduced and described in Chapter 2, is a deliverable for the More Operative Robust and Extended GNSS Open Service Signal Integrity Protection (MORE GOSSIP) project, funded by European Space Agency (ESA). While there are other more sophisticated tools for GNSS simulation and evaluation, the community is still missing a platform to test security aspects of GNSS. The tools we developed at the University of Padova are indeed security-oriented and serve as a powerful validation tool for most of the research results presented in this thesis.

Chapter 2

Security-oriented GNSS software package

A significant part of our work involved the design and implementation of a GNSS software simulator and receiver for the validation of the proposed security mechanisms. There are several online tools that allow to simulate the GNSS signal for a specific receiving position and time, or to simulate the reception of a GNSS signal on a receiver with specific parameters. However, an open source tool was not found that allows to integrate, evaluate and test security mechanisms for the Galileo constellation. Moreover none of the available tools allows to generate both Galileo and GPS signals. For the above reasons we designed our own software package, leaving room for the integration of any security feature and evaluation tools. The software package is composed by:

A GNSS signal simulator in C++ for Windows / MacOS: this software can simulate both GPS and the Galileo signals. The software takes as input the receiver's position and time, the ephemeris file and the operation mode parameters (nominal mode or attack evaluation) and outputs the I/Q samples of the baseband signal. It has the capability to work in real time together with a Software-Defined Radio (SDR) that modulates the samples at the GNSS central frequency (roughly 1.5 GHz) and a ublox receiver for evaluation. The main novelty of the signal simulator is the capability of simulating data and signal layer attacks such as navigation message tampering, SCER attack, Forward Estimation Attack (FEA), but also Denial of Service (DoS) attacks such as the one described in chapter 4.

A GNSS software receiver (Python 3): this software receiver takes as input the I/Q baseband samples (as those generated by the signal simulator). It outputs results at several levels of the receiver's blocks: the statistical distribution of the samples values, the acquisition and the tracking matrix with the code delay and carrier phase, and the user's position and time. A mode was inserted for testing different channel coding schemes for Galileo's navigation message. Moreover an interface with the simulator allows to couple the two pieces of software in order to perform a *realistic SCER attack*. The software receiver is capable of receiving both GPS and Galileo signals.

The software package has been validated and will be delivered to ESA as part of the More Operative Robust and Extended GNSS Open Service Signal Integrity Protection

(MORE GOSSIP) project. In the following sections the two software components are briefly described, together with the implementation of the realistic SCER attack involving both simulator and receiver, as an example of the software security features.

2.1 GNSS signal simulator

The GNSS signal simulator has been implemented in C++ in order to allow for a modular structure while maintaining fast execution times. The software is based on an existing open source GPS-only signal simulator [20] written in C. In designing the simulator the following choices have been made with the aim of building a simple, modular architecture, maintaining low execution time and allowing for easy addition, especially of new security features.

- In order to allow for real time signal generation the entire Doppler and pseudorange profile of the simulation is pre-computed for each satellite, given the prior knowledge on the user's motion and satellite ephemeris. This choice allows to replace the frequent Doppler update operation with a faster table lookup, thus removing the most critical bottleneck for the execution time.
- The key object in the simulations is the SV object, which can be chosen as either a Galileo or GPS SV. In order to efficiently represent an attack scenario we chose to have three different types of SV coexisting in a simulation: the legitimate SVs, as seen by the victim receiver, the attacker's SVs, as seen by the attacker and the false SVs, as seen in the spoofed position (controlled by the attacker).
- In order to guarantee the separation of the legitimate signal from the spoofing signal produced by the attacker, we produce both outputs in a parallel fashion and mix them at the last step of the simulation, before modulation.
- Parallel execution is made possible thanks to the "thread" library. All the operations relative to the signal generation are executed in a parallel fashion among different satellites, allowing for real time execution thanks to a more efficient usage of the CPU.

A simple scheme of the simulator is represented in Fig. 2.1. The whole simulator package is composed by roughly 10000 lines of code, the 60% of which have been written from scratch by our research group. The remaining part is based on [20], even though the whole architecture was heavily modified and extended. The Ph.D. candidate has contributed to the design and implementation of the whole simulator, actively coding roughly 30% of it. The most relevant libraries that were imported and most heavily exploited are the `thread` and `mutex` from `std`, to handle all the architecture for real time execution; the `mbedtls` library was exploited for all cryptographic functions, while part of the `boost` library was used to handle real time input commands and the execution options.

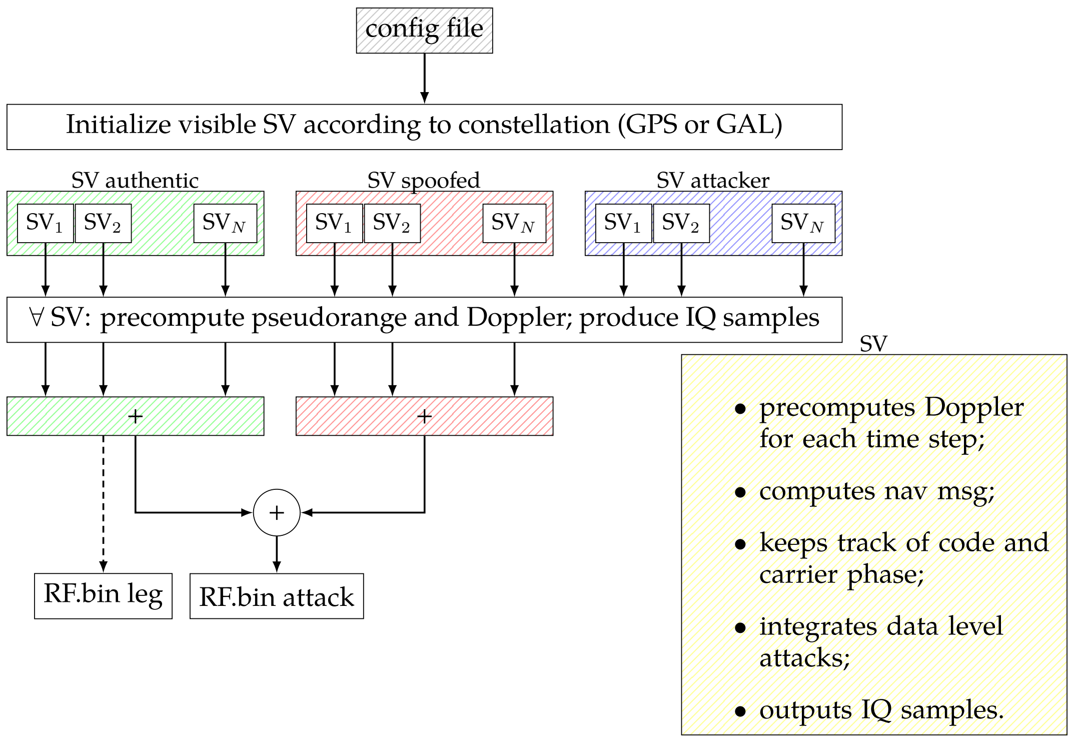

config file
Initialize visible SV according to constellation (GPS or GAL)
SV authentic
SV spoofed
SV attacker
SV_1 SV_2 SV_N
∀ SV: precompute pseudorange and Doppler; produce IQ samples
+
+
+
RF.bin leg
RF.bin attack
SV
• precomputes Doppler for each time step;
• computes nav msg;
• keeps track of code and carrier phase;
• integrates data level attacks;
• outputs IQ samples.

Figure 2.1: Block representation of the C++ GNSS software signal simulator.

2.2 GNSS software receiver

The GNSS software receiver takes as input the I/Q samples generated either from a software simulator or dumped from a receiver front end. As in the typical design, the receiver is composed by few main blocks:

Frequency analysis block, that takes as input the I/Q samples and computes the FFT, revealing the frequency representation of the signal. The presence of anomalies such as narrowband interference can be revealed in this early phase.

Acquisition block, that takes as input the I/Q samples and performs a correlation operation between the signal and the local replica over the whole 2-dimensional search space (Doppler frequency and code delay) for all satellites in parallel. The output of this block is an indicator for the presence of each satellite signal, the rough Doppler frequency and the code delay. This output can be saved in memory in order to skip the acquisition phase when re-processing the same signal. The user has control over the coherent integration time and the number of non coherent integration periods, as well as the thresholds to declare a signal as present.

Tracking block, that takes as input the I/Q samples, tracks the Doppler frequency and the carrier phase through the FLL and PLL and the code delay through the DLL. The user has control over the loops parameters, that can be adjusted to follow specific receiver dynamics. For Galileo signals the wipe off of the secondary code allows to increase the coherent integration time from 4ms to 100ms, potentially improving the performance of the tracking. This blocks also has the capability of working in parallel across different satellites in order to decrease the execution time. The tracking output

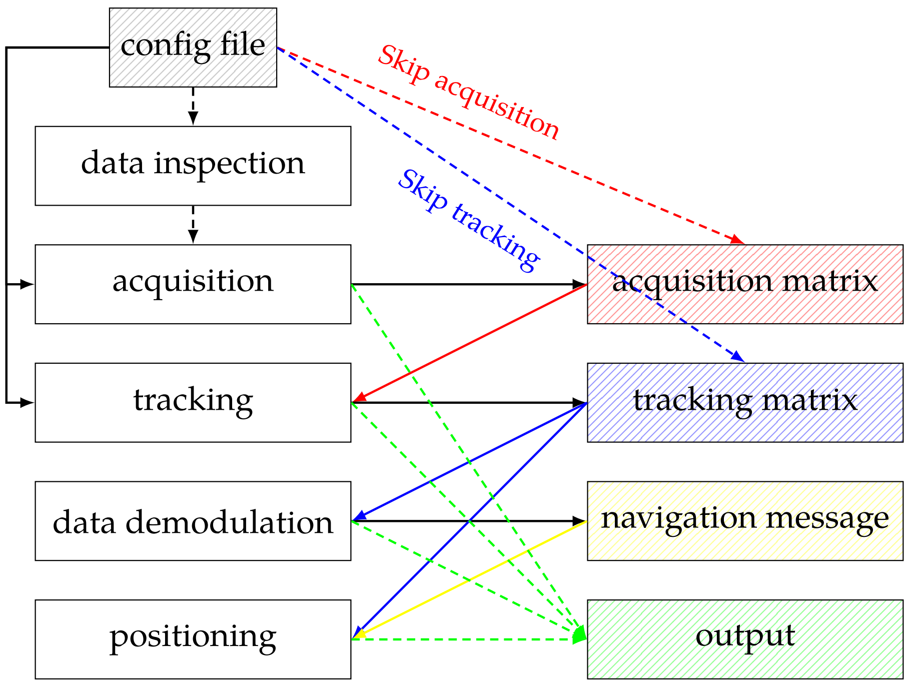


Figure 2.2: Block representation of the Python GNSS software receiver.

are the data symbols delimiters and the carrier phase and frequency for the detected satellite signals. This output can be saved in matrix form in order to skip the tracking block when processing the same signal twice.

Data demodulation block, that takes as input the output of the tracking block and the I/Q samples. the symbol values are obtained thanks to the synchronization pattern at the beginning of each word and the navigation message fields are parsed to retrieve all the relevant parameters for the positioning block. An integrity check against ill formed navigation message is inserted as suggested in Chapter 7.

Positioning block, that takes as input the navigation message parameters and the tracking block output to calculate a rough estimate of the receiver's position.

As the receiver is also security oriented, it is designed to be used mainly for assessing the effects of signal and data layer attacks on each of the basic processing blocks. A block diagram of the receiver is represented in Fig. 2.2.

The whole receiver package is composed by roughly 5000 lines of code designed and written by our research group. The Ph.D. candidate contributed to the design of the whole package, actively coding roughly 30% of the software receiver. The software simulator exploits [21], that contains tools for GNSS code generators, acquisition, and tracking, in the python/numpy environment. These tools in turn exploit the Numba JIT compiler for speeding up computations in the code generators and the Numerical Control Oscillator. The `multiprocessing` library was used to speed up execution and the `socket` library was exploited for allowing communication with the GNSS signal simulator in the scope of attack implementations such as that described in Section 3.6.

2.3 Security features

The software allows to simulate several attacks, most of which are carried out inside an SV object, while others act outside, on the signal superposition, such as the narrow band or chirp jamming and the Gaussian noise jamming.

Continuous wave jamming where a sinusoidal wave with predefined frequency and amplitude is added to the signal superposition.

Chirp jamming which implements a classic continuous wave jammer which sweeps a band around the central frequency with selected rate. A variant is implemented where the frequency drift at each step is a Gaussian random variable, making the interference harder to follow and filter.

Low power jamming that is a more sophisticated DoS attack, proposed in chapter 4. This attack is performed inside each SV object as it targets each signal individually.

Signal generation attack where no form of authentication is present and the attacker can simply reproduce the signal for an arbitrary position and time.

SCER attack as described in Chapter 3, where navigation data are supposed to be authenticated and the attacker needs to estimate the unpredictable bits. The effect of this estimation is modeled as additive Gaussian noise with zero mean and time-varying standard deviation. The simulator computes the shift the attacker has to impose to each signal in order to force an arbitrary spoofed position into the victim receiver, as outlined in Sec. 3.3.

FEA where the attacker exploits the redundancy of the convolutional code in Galileo's navigation message in order to guess some of the otherwise unpredictable bits, as described in Chapter 3. This attack assumes the implementation of NMA at the system's side.

Combined attack which puts together FEA and the SCER attack, as described in Chapter 3. FEA is used to predict the value of some of the authentication bits, while SCER is computed on the remaining unpredictable bits.

Real world SCER attack where the signal simulator and the receiver are connected and interact to perform the estimation by processing the legitimate signal *while forging the spoofing signal*. Instead of simulating the effect of the SCER attack by adding Gaussian noise to the estimated samples, the whole estimation process is implemented at the receiver's side and the estimated samples are sent back to the signal generator.

Several of the theoretical results obtained in this thesis were validated or complemented through the use of this software package.

Chapter 3

Self-spoofing attacks on GNSS signals protected by NMA

This chapter investigates the threat of *self-spoofing* through the well-known SCER attack, assessing its feasibility and limitations on GNSS signals that implement NMA. Even though the focus is on Galileo, the analysis can straightforwardly be extended to any other GNSS that aims at exploiting symbol unpredictability for anti-spoofing purposes. Even though the SCER attack has been previously addressed in the literature, an assessment on its performance at symbol level was missing. This analysis, published in [22], investigates the feasibility and limitations of a more complex attack, combining SCER with a data level estimation attack that leverages convolutional codes.

Section 3.1 introduces the design drivers and approaches to NMA, stressing the challenge of combining data and signal level authentication in the same security mechanism. Section 3.2 lists the most relevant types of spoofing threats that target NMA-protected signals and the motivations behind the investigations of the self-spoofing threat.

3.1 Navigation Message Authentication

A cryptographic mechanism that allows a user to autonomously assert the authenticity of the GNSS navigation message and protects its cryptographic integrity is commonly referred to as a NMA scheme. The aim of such a scheme is to prohibit an adversary from broadcasting counterfeit GNSS signals that contain invalid or incorrect navigation data. An efficient NMA scheme must effect a trade-off between multiple aspects, such as: security, availability, communication overhead, robustness (e.g., to channel errors and data loss), timeliness, and receiver resources [23]. In the literature several proposals for NMA scheme are available: from traditional asymmetric-based scheme such as Digital Signature Algorithm (DSA) and its many variants [24], to hybrid schemes tailored to GNSS such as TESLA [6] and SigAm [11].

In terms of fulfilling its primary task of protecting the integrity of the navigation message, a generic NMA scheme might simply be characterized by its key features, including the number of cryptographic bits inserted in the message, the equivalent security of the scheme, and the period of time over which a complete signature is broadcast. The equivalent security of the scheme indicates the difficulty in performing a brute-force attack on the underlying cryptographic primitives. The number of cryptographic bits inserted will directly influence the availability of the scheme, based on the ability of the receiver to correctly recover all of the bits, and is generally proportional to the equivalent security of the scheme. The period over which the data is broadcast will influence the latency experienced by the receiver in asserting the authenticity of the navigation data, and should therefore be commensurate and aligned with, the period over which the protected navigation data is broadcast.

When considering the indirect use of the NMA data as a mean of anti-spoofing, or range protection, one more feature must be considered: the conditional Shannon entropy of the cryptographic data given the previously transmitted messages. If the data are unknown at the time of broadcast, then one might assume that the adversary must first observe the genuine signal, before creating a counterfeit one. The likelihood of an adversary simply guessing the true value of the cryptographic data is related to the number of considered bits, and the *a priori* probability of guessing each bit. A common measure of this is the guessing entropy. If such entropy is high, then it is very unlikely that the adversary can readily produce a counterfeit signal, without first observing the genuine one. Of course, once the genuine signal is observed, a perfect replica can easily be made. This fact constrains the degrees of freedom of the counterfeit signals, in that they might only be broadcast in delay with respect to the genuine signals. It might be argued that this itself represents some defence against spoofing.

According to the Galileo Open Service, Service Definition Document (OS SDD, issued 1.1, May 2019) *The maximum nominal broadcast period of a healthy navigation message data set is currently 4 hours*. However, in order to bound the maximum error, GNSS ephemeris are considered valid over a period of two hours. Therefore the same digital signature may be repeatedly broadcast in parallel with the corresponding ephemeris data. Once the signature has been observed once, then the genuine signal becomes entirely predictable, but this fact does not impact the message integrity in any way. On the contrary, having a navigation message that does not change frequently, remaining identical over multiple re-transmissions, benefits usability. The message can indeed be recovered piece-wise over time, providing high robustness against channel errors. The same considerations hold for the digital signature: updating it at a higher frequency than the message itself impacts usability and does not benefit data level security. Extending the design of NMA to cover anti-spoofing purposes, on the other hand, requires making the digital signature unpredictable and time-varying at each transmission, forcing the receiver to recover each signature entirely.

As such, enabling some level of anti-spoofing via the insertion of high entropy, or unpredictable bits, in the navigation message necessarily comes at the cost of the availability of the navigation message authentication service. It is perhaps prudent, therefore, to weigh the true benefits of this anti-spoofing mechanism against its cost.

3.2 Spoofing attacks against NMA protected signals

Navigation message authentication is assumed to provide data level assurance, but the same does not hold for the signal in its entirety. On one hand NMA makes it impossible for an attacker to generate the navigation message and thus the GNSS signal a priori. On the other hand the attacker has other options that involve observing and exploiting the authentic GNSS signal and transmitting the forged one with minimum possible delay. The following spoofing attacks are based on the observation of the legitimate signal and affect the GNSS ranging information while leaving the navigation message untouched:

- *Meaconing:* the simplest attack is the retransmission of the whole signal as received by the attacker. This attack deceives the receiver into computing the PVT corresponding to the spoofer's antenna coordinates. On one hand this attack is straightforward to carry out, but on the other hand it is also easier to detect due to the time jump imposed at the receiver's side. Moreover, the attacker is not allowed to impose an arbitrary position onto the legitimate receiver.

- *Selective delay:* in order to induce an arbitrary PVT on the receiver, the attacker needs to separate the signals originated from different SVs. Only then a new signal can be forged by re-combining the signal components with appropriate relative delays, so that the computed ranges at the receiver' side will be those of the spoofed position. In order to achieve this signal separation two approaches are available. The attacker can either exploit high gain antennas pointing at each SV or separate the signals as any receiver does, exploiting the orthogonality of the PRNs. The latter approach requires tracking the ranging signal and integrating it over the whole PRN period, forcing the attacker to introduce a further delay in the forged signal equal to the duration of a whole PRN, causing a time jump that in the victim receiver. In [25] an approach is proposed to trade off time delay with symbol estimation noise. In this attack, commonly referred to as SCER [25, 26], the attacker refines its estimation of the unpredictable symbol at each received sample and can decide to send the spoofing signal at any time before the end of the symbol (e.g., potentially mounting 0-delay attacks).

- *FEA*: The Forward Estimation Attack exploits the forward-error correction (FEC) used in some GNSS signals as a means of producing counterfeit signals in a non-causal manner [27, 28]. The inherent redundancy of coded navigation messages is leveraged to predict the value of later symbols in a codeword, based on observations of the earlier symbols. Depending on the coding rate and the *a priori* knowledge of the navigation data, the adversary can create counterfeit signals *in advance* with respect to the observation of the genuine signal. This attack involves guessing some of the unpredictable symbols, trading off success probability for time delay. However the coding redundancy ensures that, even when several of the early symbols have been guessed incorrectly, the counterfeit symbols will decode to the correct navigation data with non negligible probability, depending, to some extent, on the victim decoder.

- *State Modeling Attack (SMA)*: The State Modeling Attack targets a set of spoofing detection mechanisms that exploit correlation based signal verification across the unpredictable navigation data symbols [28]. As the correlation process is additive and the signal-to-noise ratio is very low, the legitimate receiver has no way to tell apart the contribution of different symbols in the test statistics, or even detect variations of the instantaneous power inside a single symbol. As the test statistic cannot distinguish between energy accumulated in one symbol over another, the adversary can modify the instantaneous power of later symbols, based on the *a posteriori* knowledge of how many of the earlier symbols were guessed correctly. This strategy can significantly reduce the effectiveness of the anti-spoofing scheme.

In the literature the described attacks are analysed separately, supposing that the attacker uses the same attack strategy for all the spoofed SVs, and for the entire attack duration. Although this allows to assess the performance of the particular attack under analysis, this does not represent a realistic and sophisticated attacker aiming at optimizing its strategy. Indeed, depending on multiple factors such as the constellation geometry seen by both the attacker and the victim receiver, and the local environment, the attacker could select a different strategy across distinct PRNs. For instance, some signals will be received with higher C/N_0 with respect to others, requiring a shorter integration time for the same symbol error probability. Moreover, depending on the constellation geometry and the relative distance between spoofer and attacker positions, the attacker will need to delay some signals and possibly anticipate others, imposing different timing requirements in the symbol estimation. Finally, it is worth noticing that since only a fraction of the symbols is unpredictable, the attacker can exploit its *a priori* knowledge on the other symbols.

3.2.1 Self-spoofing attacks

Self-spoofing is a particular case of spoofing attacks where the GNSS receiving equipment is assumed to be under control of the adversary. Such scenarios represent some of the most demanding requirements for protection against GNSS spoofing. Two representative applications were considered in the definition of a baseline spoofing scenario and its respective assumptions: satellite-based Vessel Monitoring Systems (VMS) and smart digital tachographs. Both applications have the following characteristics in common:

- Applications are used for enforcement of European Commission (EC) regulations (e.g., to ensure fair competition, protection of fishing stocks, road safety, minimum working conditions for professional drivers, etc.);
- Applications are backed by EC implementing regulations that require or prescribe mechanisms to prevent falsification of position;
- Self-spoofing is the predominant attack scenario, where the adversary has an incentive to attack the system in order to obtain advantage (e.g., access to closed fishing areas, exceeding daily driving hours to obtain a competitive advantage, etc.);
- Applications have access to communication networks for reporting non-compliance (e.g., Satellite communications, Direct Short Range Communications (DSRC)).

Satellite-based VMS' were introduced to protect fisheries from illegal fishing, using GNSS for enforcement. Requirements for VMS' are largely driven by regulation [29], specifying acceptable position errors and associated confidence intervals, the contents and frequency of transmissions to a Fishing Monitoring Centre (FMC) that allow authorities to react in a timely manner to non compliant behaviour.

The regulation notes that "Member states shall adopt the appropriate measures to ensure that the satellite-tracking devices do not permit the input or output of false positions and are not capable of being manually over-ridden." Today, such measures include the use of tamper-resistant hardware and cryptographic mechanisms to provide protection against vessel owners attempting to tamper or interfere with the device; however, it is only a matter of time before malicious owner resort to GNSS spoofing as a threat to the present enforcement mechanism. The return on investment for defeating the VMS and illegally fishing in closed areas can be significant and can substantially outweigh the costs associated with conducting a spoofing attack. In fact, data from the VMS has been accepted as evidence in court against offences related to unlawful entry or illegal fishing in closed areas. On the other hand, evidence related to tampering or interfering with a VMS transponder and provision of false information to the relevant fisheries administration has also been accepted [30], making such attack less attractive.

Digital tachographs record the activities of professional drivers including rest and driving hours, to ensure compliance with EC regulations [31], increasing road safety, ensuring minimum working conditions and guaranteeing fair competition for EU transport companies. EC regulation [32], prescribing requirements for construction, testing, installation, operation and repair of tachographs and their components addresses the use of GNSS, remote early detection of possible manipulation or misuse, interfaces with intelligent transport systems and security mechanisms.

To address risk of tampering and falsification of GNSS position, the regulation prescribes error reporting for discrepancies in time between GNSS and the vehicle unit, and discrepancies in motion determined from a motion sensor (e.g., tachometer) and GNSS-estimated motion. Moreover, the specification of the GNSS receiver states that "The GNSS receiver shall have the capability to support Authentication on the Open Service of Galileo when such service will be provided by the Galileo system and supported by GNSS receiver manufacturers.". While this function is not available today, a recent invitation to tender [33] by the European GNSS Agency, targets the development, supply and testing of a Galileo Open Service Authentication User Terminal, including a prototype of a terminal meeting the smart digital tachograph application requirements.

The following analysis considers as a baseline a GNSS receiver implementing defences based on the utilisation of Galileo Open Service authentication as described in [33]. Future work may consider an extension of this receiver with an external motion sensor (i.e., tachometer), implementing the time and motion discrepancy checks prescribed in the implementing regulation.

3.3 Spoofing attacks in the real world

Fig. 3.1 shows the considered spoofing scenario, consisting of two entities: the spoofer and the victim receiver.

It is assumed that the self-spoofing user is able to smoothly transition the receiver from the condition where it is illuminated by genuine signals only, to that where it is

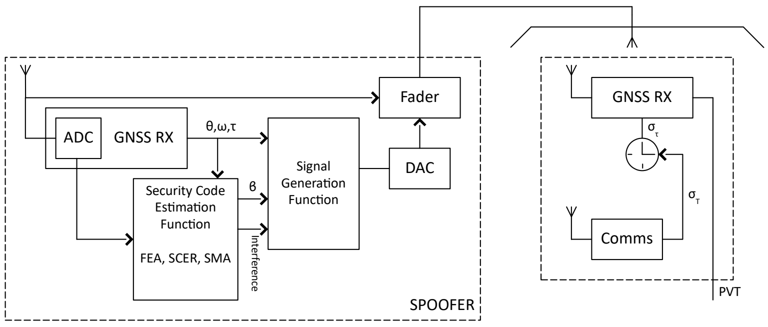


Figure 3.1: Considered self-spoofing scenario.

illuminated by forged signals only. The transition may either be smooth (carry-off) or abrupt (suddenly covering the antenna); the latter case may be disguised as a natural outage of GNSS (obscuration by artificial landmarks, e.g., a bridge). The fader is the element responsible for shaping this transition. Once the receiver is in a covered state, it only observes inauthentic signals.

The objective of the spoofer is to ensure that the receiver cannot reliably assert that these signals are not genuine. In the absence of NMA, the spoofer could simply generate consistent GNSS signals with modified reciprocal time delays. On the other hand, the unpredictability of the cryptographic information introduced by NMA limits the spoofer's freedom. In order to generate ranging signals that can pass the cryptographic verification, the spoofer exploits its GNSS receiving equipment (possibly comprising multiple antennas) to track the Signal-In-Space (SIS) and feed the Security Code Estimation function. This is the fundamental block in the spoofer, which performs the estimation of the cryptographic data and feeds the signal generation block based on the attack strategy that minimizes the detection probability. The digital spoofing signal is finally converted to RF and combined with the legitimate signal before being transmitted to the victim receiver.

The receiver is a traditional single frequency, single antenna GNSS receiver. We assume that the receiver has sporadic access to the network that can be exploited for achieving coarse time synchronization.

A representation of the general spoofing scenario is given in Fig. 3.2. Since in typical NMA proposals the authentication data are broadcast jointly with the navigation message, the unpredictable symbols are interleaved with predictable ones. For instance, in [33], the unpredictable symbols are contained only in the odd Galileo INAV half-page, while the even half-page is completely predictable. Moreover, even in the odd INAV half-pages not all symbols are unpredictable. For this reason, the attacker could guess part of the unpredictable data before observing it, thanks to FEA.

Let us denote by t_p the delay introduced by the signal processing operations together with the hardware delay of antennas and cables, and by t_d the propagation delay due to the distance between the spoofer transmitting antenna and the victim antenna, which is negligible in a self-spoofing scenario. $\Delta_i(t)$ indicates the difference between the geometric ranges of the spoofer and the falsified position with respect to a SV$_i$.

For those satellites whose signal arrives earlier to the spoofer antenna than to the false position ($\Delta_i(t) > 0$), the attacker can easily emulate the legitimate signal, adding the

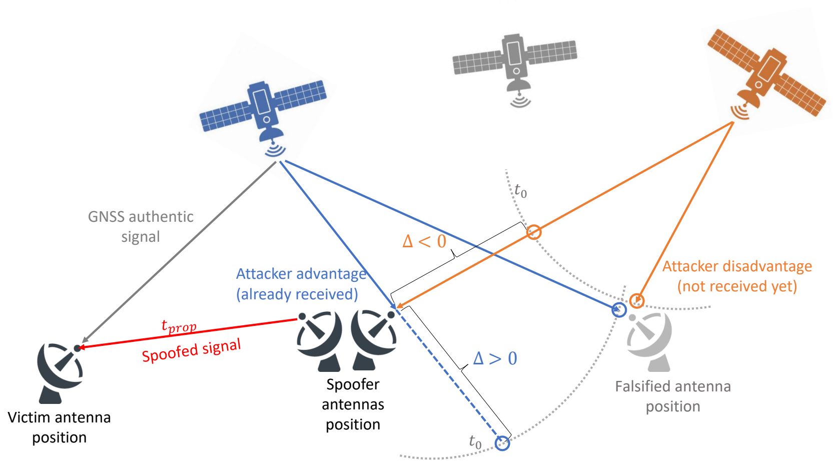

GNSS authentic signal
Attacker advantage (already received)
$\Delta < 0$
t_0
Attacker disadvantage (not received yet)
t_{prop}
Spoofed signal
Victim antenna position
Spoofer antennas position
$\Delta > 0$
t_0
Falsified antenna position

Figure 3.2: Pictorial representation of the spoofing scenario.

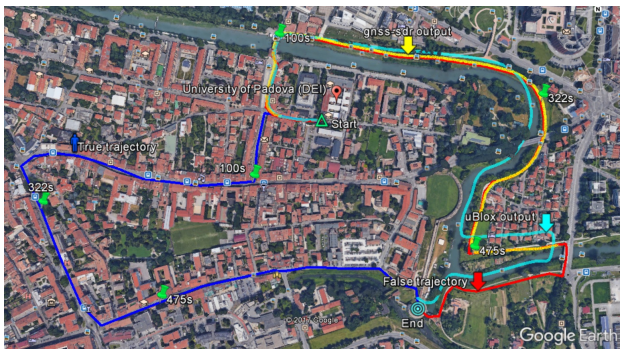
appropriate delay. The higher this delay, the better for the attacker, since more energy can be accumulated, thus obtaining a more reliable estimate and a more accurate replica. On the contrary, some signals will likely need to be anticipated ($\Delta_i(t) < 0$), i.e., to be transmitted *before* the attacker sees the corresponding values. In this case the attacker can leverage FEA to obtain an a priori knowledge of part of the page and perform a SCER attack on the unpredictable symbols. In order to produce a spoofing signal with the correct timing, the attacker shall start the transmission even before the leading edge of the unpredictable symbol reaches his antenna. The transmission can thus be started in advance, introducing random samples at the beginning of the unpredictable symbols, then perform a zero-delay SCER attack. If the number of introduced random samples is low, the effect on the probability of correct symbol decoding by the victim receiver is negligible, as proved in the following sections.

A second option for the attacker is that of introducing a time offset t_{off} between the receiver clock and the system time. This would allow receiving at least a few samples of each symbol before transmitting the corresponding estimate. The time offset shall meet $t_{\text{off}} \geq t_{\text{p}} + t_{\text{d}} - \min \Delta_i(t)/c$. On the other hand, the time offset that can go undetected depends on the clock uncertainty of the victim receiver. The larger the introduced offset, the better the estimation performance for the attacker. As a matter of fact a large enough offset could allow the attacker to decode and transmit the symbols without performing either FEA or SCER. This makes receivers particularly vulnerable during cold- or warm-start where the receiver clock bias might be large. Even if the receiver is equipped with a reasonably high quality oscillator, e.g., with an accuracy of 100 ppb, it would still loose about 100 us within approximately 15 minutes.

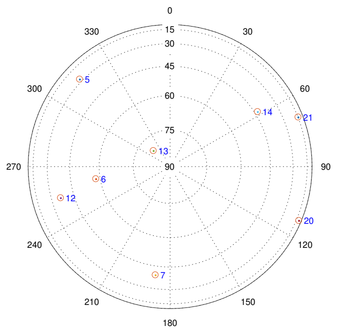
A real world example of the spoofing scenario is shown in Fig. 3.3a with the corresponding skyplot reported in Fig. 3.3b, where the SVs belong to the Galileo constellation. A self-spoofer moves, at a constant speed of 10 km/h, from the University of Padova building following the blue trajectory, while generating a falsified signal that leads the receiver to compute the PVT corresponding to the red trajectory. Fig. 3.4 represents the difference between ranges computed in the false position and in the attacker position.

The attack was successful for both the u-blox M8T and the software receiver. A difference can be noticed in the trajectory computed by the two receivers; this is probably due to the fact that we have less control on the set of satellites employed for the PVT in the u-blox receiver. In the example, it can be seen that, at $t = 100$ s, the attacker would need to anticipate by 500 ns the signal coming from PRN 5, while that from PRN 7 needs to be delayed by 833 ns. The relative movement between authentic and false position causes the time offset to be highly dynamic: conversely, in the last part of the observed time window, the signal from PRN 7 needs to be anticipated and that from PRN 5 delayed. It is worth noting that with a sampling frequency of 4 MHz, a sample corresponds to 250 ns, hence the mentioned delays correspond to just few samples.

It is worth noticing that is not necessary for the attacker to spoof all the satellites in view: the target can be limited to those SVs on which the attacker has an advantage over the false position. For example at $t = 100$ s he can spoof SVs 6, 7, 12 and 20. After some time, PRN 20 can be excluded from signal generation and replaced by SVs 5 and 13. This motivates to differentiate the attack strategy among the different SVs and over time. This approach may degrade the navigation of the victim receiver and in a typically good reception context such as VMS, this effect may be observed in metrics such as dilution of precision (DOP). On the other hand, in a more unstable context, e.g., for a moving receiver in obstructed environment it may be hard to discriminate between impaired reception conditions and a carefully crafted scenario.



(a)



(b)

Figure 3.3: Example of real world spoofing scenario. (a) The actual trajectory of the self spoofer (in blue), the intended spoofed trajectory (in red), and the computed position (in light blue for ublox receiver and in yellow for software receiver). (b) Skyplot of the SVs in view in the example spoofing scenario.

3.4 Success Probability of the Attack

3.4.1 Forward Estimation Attack

The aim of FEA is to exploit the redundancy in FEC to reduce the amount of unknown symbols in the navigation message. Let us take as an example the Galileo E1B signal component and its NMA scheme as per [33], that is transmitted using the EDBS channel contained in the odd half pages of the INAV message. Out of the 40 bits of the EBDS

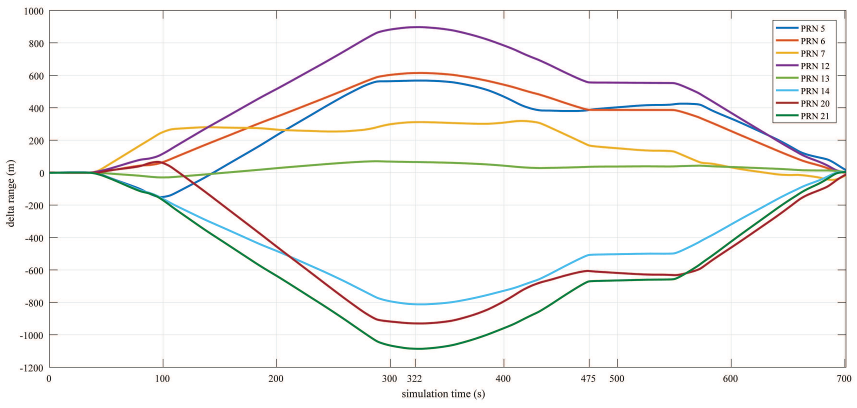


Figure 3.4: Δ values for the SVs in view in the example spoofing scenario of Fig. 3.3.

channel, 8 are devoted to key management and are considered predictable [34], leaving 32 unpredictable symbols. The INAV message is protected by a parity check consisting in a cyclic redundancy check (CRC) of 24 bits [35]. The coded symbols are then interleaved in order to increase the robustness to error bursts and fading scenarios. This causes some of the symbols of the CRC to be transmitted before the unpredictable ones. Therefore, an attacker needs to guess not only the unpredictable symbols but also the CRC, which gives N_{US} symbols to guess for each odd half-page:

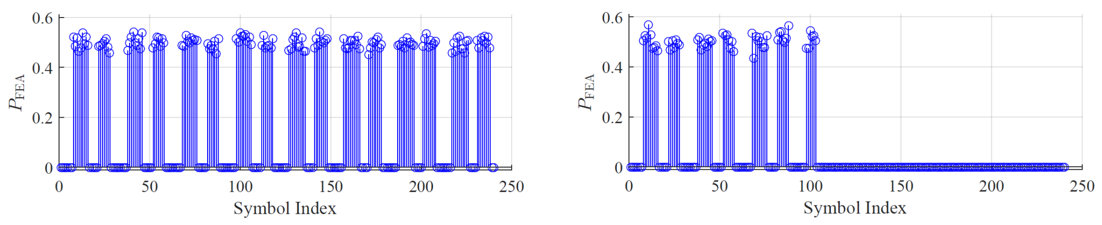
$$N_{US} = 2(N_{UB} + L_{CRC} + 2(K-1))$$

where N_{UB} is the number of cryptographic bits, L_{CRC} is the length in bit of the CRC and K is the constraint length of the convolutional code. Note that the constraint length contributes twice to the number of unpredictable symbols, as the cryptographic data and the CRC are broadcast as two groups of bits, separated by more than K bits. With the chosen parameters we have $N_{US} = 136$. While it is possible to improve the FEA trying to brute-force the CRC [36], even a basic implementation of the FEA reduces the number of unpredictable symbols to 56. By extending FEA to brute-force search across the full 24-bits CRC, the number of unknown symbols can be reduced to 51. The number of resolved symbols can be tradeoff with computational complexity, allowing to impose a more realistic bound to the attacker's capabilities. Fig. 3.5 shows the result of FEA on

Attack paramters	N_{US}
no FEA	136
FEA with no CRC brute-force	56
FEA with 14-bit CRC brute-force	54
FEA with 20-bit CRC brute-force	53
FEA with 24-bit CRC brute-force	51

Table 3.1: Number of symbols (N_{US}) of the odd-pages of the INAV message that remain unpredictable after applying one of the feasible prediction algorithms.

the 240 interleaved symbols of the odd-half page. It can be noticed that the prediction algorithm is unable to get any advantage in the guessing probability of the first half of



(a) Without FEA or CRC brute-force (b) FEA and 14-bit CRC brute-force

Figure 3.5: Probability of error when guessing each symbol of the navigation message odd-page assuming without FEA (a) and with FEA and a brute-force search of the last 14 bits of the CRC (b)

the unpredictable symbols. However, after accumulating approximately half of the 240 symbols, FEA allows to correctly decode the remaining symbols with probability equal to 1.

The attack strategy can be optimized by resolving half of the unknown symbols with FEA and performing a SCER attack on the remaining ones. This approach reduces the number of symbols that are actually estimated on the fly and replayed. Therefore a spoofing detection metric that leverages the attacker's estimation algorithm can only rely on 51 symbols, corresponding to 204 ms, every two seconds. This means that just about 10% of the signal can be used for anti-spoofing. The attacker can exploit a window of 1.56 s of completely predictable symbols every two seconds, inducing a small drift in the local clock of the receiver that accumulates over time.

The FEA attack was implemented and tested with the C++ signal simulator developed in the University of Padova, obtaining a correct page decoding rate, P_c, of around 1%, when 32 unpredictable bits are employed, as in [36]. This result translates into a security level of $-\log_{1/2}(P_c) = 6 - 7$ bits per page, with respect to the intended 32 bits. It is worth noting that this result was obtained by testing the 32-bits NMA field on i.i.d. random bits with equal probability. When the test was repeated with a simple ECDSA signature implementation of NMA, the result showed that fewer bits of security were left after the FEA attack, due to the fact that some parts of the signature were not completely unpredictable. The signature padding, accessory information fields and identifiers for the signature chunks are all examples of data that have a known structure and are not full-entropy.

3.4.2 Security Code Estimation and Replay Attack

This section aims at evaluating the effect of a SCER attack on signals that are protected by NMA. We take as a metric of the attack feasibility the probability of correct symbol demodulation at the receiver's side. The attack is considered feasible if the NMA protected signal can be spoofed with a non negligible probability of correct symbol demodulation.

The SCER attack aims at transmitting the estimated symbols from the legitimate signal with minimum delay. Instead of accumulating all the symbol samples to obtain a reliable estimate, at each received sample the adversary transmits an approximate estimation of the symbol based on all the samples received up to that instant. In [25] and [26] several estimation strategies were defined; this analysis assumes the use of the maximum a posteriori probability (MAP) estimator. This attack causes the first transmitted samples to be noisy, since they are computed by observing just a small fraction of the legitimate symbol. However, as soon as more energy is accumulated from the legitimate symbol, the transmitted values will become more reliable and the probability of sending a correct sample will exponentially increase.

Let us analyze the probability of decoding errors on the symbol at the receiver side, comparing the results obtained in the following cases:

- the receiver is decoding data from the authentic signal;
- the receiver device is subject to a SCER attack.

In the former case this probability will depend on the value of C/N_0, taking the form:

$$P_{e,L} = \frac{1}{2} \operatorname{erfc}\left(\sqrt{T_s C_L / N_0}\right)$$

where T_s represents the symbol period and C_L the power of the received legitimate signal power.

In the latter case it is expected that the effect of the non-reliable initial estimates will increase the probability of decoding error. Since at time t, the attacker sends its estimate based on the previously observed samples, this is equivalent to the legitimate case, for a symbol with duration t, and thus:

$$P_{e,A}(t) = \frac{1}{2} \operatorname{erfc}\left(\sqrt{t C_A / N_0}\right)$$

where $P_{e_A}(t)$ represents the probability of decoding error *for the attacker* at time t (at the attacker's side *the reception* of a symbol is supposed to start at $t = 0$). C_A represents the received signal power at the attacker's side. The victim will correlate the signal received from the attacker with its local replica, obtaining a positive contribution for the samples that were correctly estimated by the adversary (with probability $1 - P_{e_A}(t)$), and a negative contribution for the others (with probability $P_{e_A}(t)$). The energy of the correlation on the whole symbol is thus given by:

$$\begin{aligned} E_A &= \int_0^{T_s} \left[C_S(1 - P_{e_A}(t)) - C_S P_{e_A}(t)\right] dt \\ &= C_S \int_0^{T_s} \operatorname{erf}\left(\sqrt{t C_A / N_0}\right) dt \,. \end{aligned}$$

where C_S is the spoofed signal power received by the victim. In the above formulation the unpredictable symbols are assumed to take values in $[-1; 1]$ with P(-1) = P(1) = 0.5. Moreover, the attacker is assumed to transmit a spoofing signal that is received by the victim at the same power $C_S = C_L$, thus removing one degree of freedom for the attacker. Several detection mechanisms exploit the power level of the received signal [37,38], but this extension is left for future work.

In general the spoofed position is likely to have a significant offset with respect to the actual position of the user's device, therefore the attacker will have to either anticipate or delay the signal from each SV. Delaying a signal gives the attacker more time to refine its estimate before sending the first samples of the symbol. Let T_d represent the time delay introduced with respect to the attacker's received signal. The attacker will now accumulate the legitimate signal for T_d seconds, sending it's first estimated sample at $t = T_d$. For a duration of $T_s - T_d$ the attacker experiences an increase in the reliability of its estimate, which saturates after T_s seconds, when the final sample is received. Until the end of the transmitted symbol, $T_s + T_d$, the attacker will send the last estimated samples, all equal to the one sent at T_s. Thus, the probability of an incorrect estimate is:

$$P_{e,A}(t) = \begin{cases} 1/2\,\text{erfc}\left(\sqrt{tC_A/N_0}\right), & \text{for } T_d \leq t \leq T_s \\ 1/2\,\text{erfc}\left(\sqrt{T_s C_A/N_0}\right), & \text{for } T_s \leq t \leq T_s + T_d \end{cases}$$

On the contrary in case the signal needs to be anticipated of T_a, for $-T_a \leq t \leq 0$ the attacker has to guess the symbol:

- the spoofer can flip a coin and choose either 1 or -1, then stick to the chosen value until he starts receiving the legitimate signal, gaining some information about the actual value of the symbol;
- the spoofer can flip a coin at each sample, transmitting a variable outcome. On average around half of the samples will be correct;
- the spoofer can transmit nothing until he starts receiving the legitimate symbol.

One important consideration is that the signal must be trackable by the target receiver. That is, if the time advance the adversary needs to introduce is too long, (e.g., a few ms), then the receiver may experience tracking errors or perhaps even loss of lock. The second and third options are valid for short time shifts (e.g., 10-200 μs), but may not work for longer ones (e.g., 1-10 ms). The spoofing signal needs to exhibit good correlation with the local replica in order for the target receiver's PLL and DLL to work properly. For short range time shifts, of the order of some tens of μs, the result is the same in all three cases; thus the following analysis assumes the use of the second strategy for the attacker.

Starting from $t = 0$ the attacker starts accumulating samples from the real signal, thus switching from a random guess to the actual SCER estimation. Then the probability of incorrect estimate is:

$$P_{e,A}(t) = \begin{cases} 0.5, & \text{for } -T_a \leq t \leq 0 \\ \frac{1}{2}\,\text{erfc}\left(\sqrt{tC_A/N_0}\right), & \text{for } 0 \leq t \leq T_s - T_a \end{cases}$$

Since some signals will need to be anticipated (case of T_a) and some others delayed (case of T_d), let us synthesize the analysis by considering a more general time shift of Δ, where:

- in case of $\Delta < 0$ the signal will be anticipated with $T_a = -\Delta$
- in case of $\Delta \geq 0$ the signal will be delayed with $T_d = \Delta$

The result is:

$$E_A = \begin{cases} C_S \int_0^{T_s+\Delta} \operatorname{erf}\left(\sqrt{tC_A/N_0}\right) dt, & \text{for } \Delta < 0 \\ C_S \int_\Delta^{T_s} \operatorname{erf}\left(\sqrt{tC_A/N_0}\right) dt + C_S\Delta \operatorname{erf}\left(\sqrt{T_sC_A/N_0}\right), & \text{for } \Delta \geq 0 \end{cases}$$

Let's now consider $C_S \int_a^b \operatorname{erf}\left(\sqrt{tC_A/N_0}\right) dt$, whose value can be obtained in closed form by performing a change of variable, $x = \sqrt{tC_A/N_0}$, obtaining

$$C_S \int_a^b \operatorname{erf}\left(\sqrt{tC_A/N_0}\right) dt = 2\frac{N_0C_S}{C_A}\left[\frac{(2x^2-1)\operatorname{erf}(()\,x)}{4} + \frac{xe^{-x^2}}{2\sqrt{\pi}}\right]\Bigg|_{\sqrt{aC_A/N_0}}^{\sqrt{bC_A/N_0}} .$$

By substituting the derivation with the corresponding integration limits inside the above equation, it is possible to obtain E_A, and from this value one can derive the probability of error on the final symbol at the victim's side as

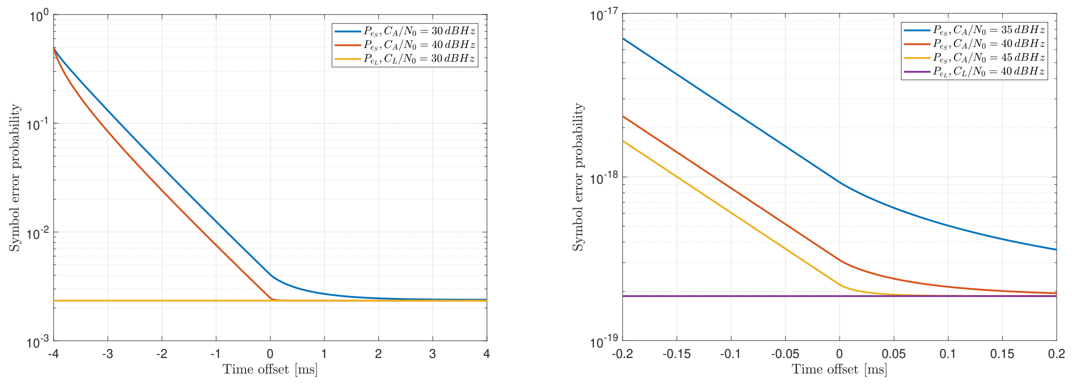
$$P_{e,S} = \frac{1}{2}\operatorname{erfc}\left(\sqrt{E_A/N_0}\right)$$

Figure 3.6a represents the probability that the victim's receiver decodes a wrong symbol with respect to the legitimate one, against the time offset imposed by the spoofer. The effect of the attack is compared with the symbol error probability in case of legitimate signal, that is only due to AWGN noise. In this case the C/N_0 at the legitimate receiver's side is $C_S/N_0 = 30$ dBHz, while that at the spoofer's side is either $C_A/N_0 = 30$ dBHz or $C_A/N_0 = 40$ dBHz. Notice that the time offset in the figure lies in the range $-T_s \leq \Delta \leq T_s$, where $T_s = 4ms$ is the Galileo E1 symbol period:

- $\Delta = -T_s$ represents an attack in which the spoofer is trying to guess the entire symbol by flipping a coin. In the figure, the corresponding symbol error probability is 0.5.
- $\Delta = T_s$ represents an attack in which the spoofer is transmitting his estimate after receiving the whole symbol, which is the best he could do in the most favourable case.

Fig. 3.6b is the analogous of Fig. 3.6a, but in the case $C/N_0 = 40$ dBHz. Notice that error probability values for the spoofed signal and the legitimate signal decrease by several orders of magnitude and so does their difference, even in case the attacker has a lower C_A/N_0 than the legitimate receiver (e.g., 35 dBHz, as shown in the figure). This effect is due to the fact that a receiver with higher C/N_0 is able to obtain a more precise symbol estimate, that is hardly corrupted by a few wrong samples. On the contrary, if the receiver's estimate is not very resilient to start with, the effect of guessing some samples will have more weight in the final error probability.

Fig. 3.7 represents the contour plot of the logarithm of the symbol error probability at the receiver's side, with varying C_A/N_0 for the attacker (Fig. 3.7a) and for the receiver (Fig. 3.7b). Notice that a high C_A/N_0 allows to lower the error probability even when

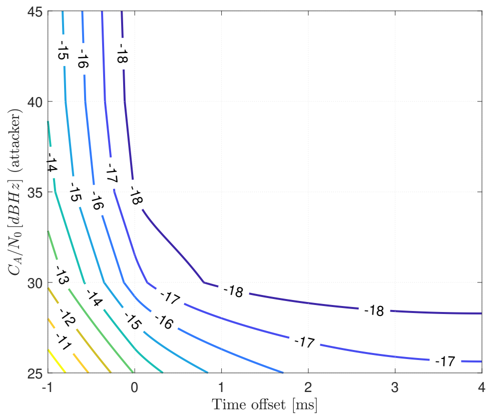


(a) (b)

Figure 3.6: Symbol error probability with respect to the time offset. The receiver C/N_0 is respectively of 30 dBHz in (a) and 40 dBHz in (b).

anticipating the signal by a few samples. On the other hand, as discussed above, a lower receiver's C/N_0 increases the difference between the symbol error probability in the attack case and in the legitimate case. As it can be seen from the figure, between the two parameters the most relevant is the C/N_0 of the receiver, that brings a greater variability in the symbol error probability with respect to C_A/N_0.

The results show an overall increase in the symbol error probability due to the SCER attack with respect to the legitimate case. When taking into account the overall effect of the combined attack (FEA and SCER), it is worth noting that for those receivers that exploit FEC an incorrect symbol does ot necessarily lead to an incorrectly decoded page. Symbol errors have indeed a non negligible probability of being corrected by FEC algorithms, which is the principle exploited by FEA itself. The difference in the symbol error probability between the spoofed signal and the legitimate one may hint at leveraging this metric for anti-spoofing purposes. As it appears from the graphics, this difference is more significant for low receiver's C/N_0, due to the fact that being so close to the decoding threshold, any negative contribution to the symbol energy can make a difference. However, working at low C/N_0 impacts the receiver performance. The variability of the channel through time has not been taken into account in this analysis, moreover it has been assumed that the receiver has perfect knowledge of its own C/N_0, which is in general not the case.


(a) Varying attacker's C_A/N_0; the receiver's C/N_0 is set at 40 dBHz.

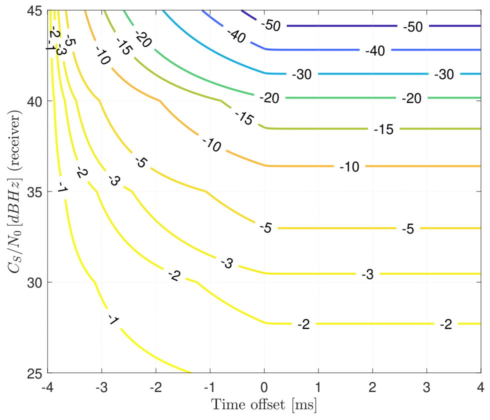

(b) Varying receiver's C/N_0; the attacker's C_A/N_0 is set at 40 dBHz.

Figure 3.7: Level curves of the symbol error probability (in the form $\log_{10}(P_e)$) as a function of the induced time offset and the C_A/N_0 of the attacker (a) and C_S/N_0 of the receiver (b).

3.5 Statistical distinguishability

The above analysis has evaluated the feasibility of the attack and its impact on the NMA cryptographic verification. In the following the attack limitations will be evaluated by considering the Mean Square Error (MSE) and the mean error energy in the symbol as a measure of the distinguishability between the authentic and the spoofed signals.

Let us denote the legitimate signal by $s(t) = d(t)c(t)\sin(2\pi ft + \varphi)$ and by $r(t) = s(t) + n(t)$ the received legitimate signal corrupted by noise. Instead, let $\hat{s}(t) = \hat{d}(t)c(t)\sin(2\pi ft + \varphi)$ denote the spoofed signal generated through the SCER attack, and the corresponding $\hat{r}(t) = \hat{s}(t) + n(t)$. We assume that the attacker is not influencing the noise level seen at the victim receiver, thus the noise term have the same distribution in both cases.

As we are interested in deriving a bound for the detection capabilities, we neglect tracking errors such as carrier and code phase errors. Let us define the MSE as

$$e(x,t) = \frac{1}{N_s}\sum_{N_s}|x(t) - s(t)|^2$$

where N_s is the number of symbols over which the measure is computed and x can be either r or $\hat{r}$. The overall mean error energy is defined as:

$$E(x) = \frac{1}{T_S}\int_{T_s} e(x,t)dt \; .$$

At the University of Padova we developed a C++ GNSS signal generator for both GPS L1 C/A and Galileo E1B and E1C signals. The constellation simulator produces baseband signals that are then processed using both GNSS-SDR [39] software receiver and transmitted using a bladeRF [40] software defined radio to a ublox M8T receiver. The software integrates cryptographic NMA and advanced spoofing capabilities, including the combination of FEA and SCER attack. The software generator was used to validate our theoretical results and assess the performances of the attack.

The probability of correct decision on the samples of a symbol is represented in Fig. 3.8 for different values of C/N_0.

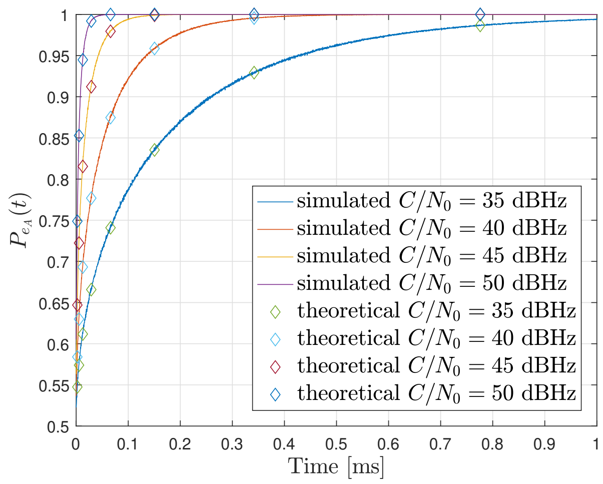
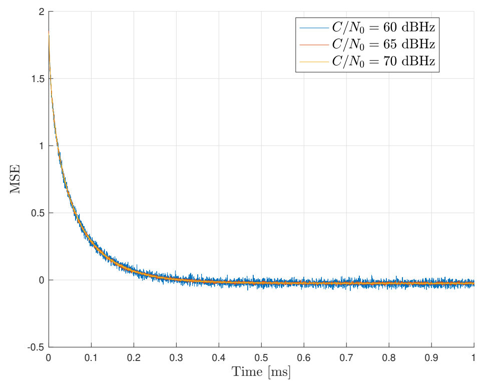


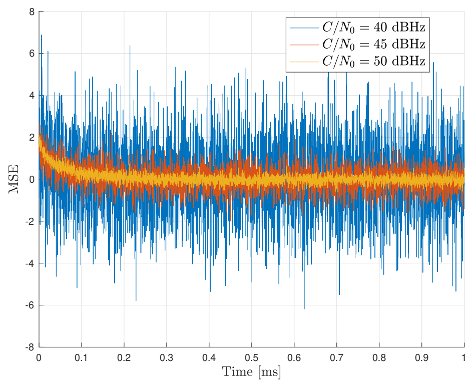
Figure 3.8: Probability of correct decision on symbol samples, averaged over multiple symbols.

Fig. 3.3a shows the result of a spoofing simulation with real-world trajectories: both the ublox M8T and the software receiver GNSS-SDR computed the false PVT solution, induced by the spoofing signal, even with moderate C/N_0. The MSE metric was evaluated on the spoofing scenario presented in Fig. 3.3a. The values are averaged over 9 SVs, with a simulation time of 400 seconds each, for a total of one hour of signals. In Fig. 3.9, the computed MSE over the symbol duration, $e(x, t)$, is reported. The signals were normalized to unitary amplitude and the mean value of the MSE was removed. Fig. 3.9a highlights the error contribution of the spoofer estimation at the beginning of the symbol. However, since it is derived under the unrealistic assumption of $C/N_0 = 60$ dBHz or 70 dBHz for the receiver, it only serves an illustrative purpose. When using more realistic values for C_A/N_0, such as in the 40-50 dBHz range, it easy to see from Fig. 3.9b that the receiver noise dominates the error contribution, while the additional errors introduced by the spoofer become negligible. This is even more evident in Fig. 3.10, where the MSE value exhibits an increase of less than 0.5% in the initial part of the unpredictable symbols with respect the legitimate case. It is worth remarking that these results are obtained by neglecting all errors sources beside the AWGN noise (e.g., PLL/DLL tracking errors, ADC saturation...).

Fig. 3.11 shows another metric, the error energy, $E(x)$, represented for ease of visualization as $10\log_{10}\left(|E(\hat{r}) - E(r)|/E(r)\right)$. The figure shows that a reasonably high distinguishability can be achieved only for unrealistic values of the legitimate receiver's C/N_0, while for a receiver in nominal conditions detecting the error contribution of a spoofer becomes really hard, due to the high noise floor. In Fig. 3.11a the MSE is computed only over the first 50 μs of the symbol, targeting the portion of the symbol that is most impacted by the spoofer and allowing for a better resolution. In order to optimize the detection capability, a proper weighting function should be designed that matches



(a)



(b)

Figure 3.9: MSE $e(\hat{r}, t)$ (after subtracting the average value) computed assuming $C_A/N_0 = 40$ dBHz for different victim receiver C/N_0 levels, averaged over one hour of signals.

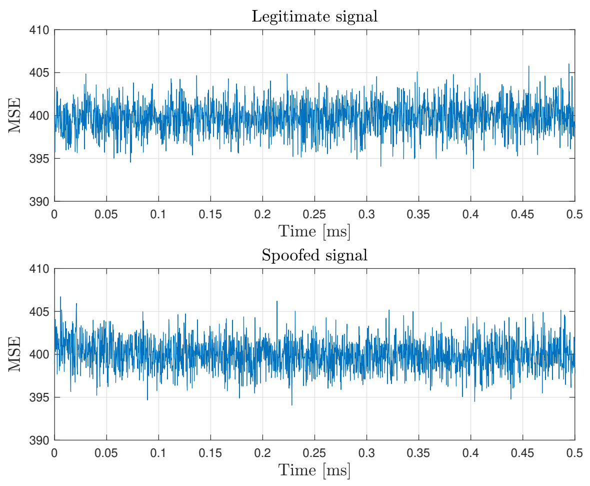
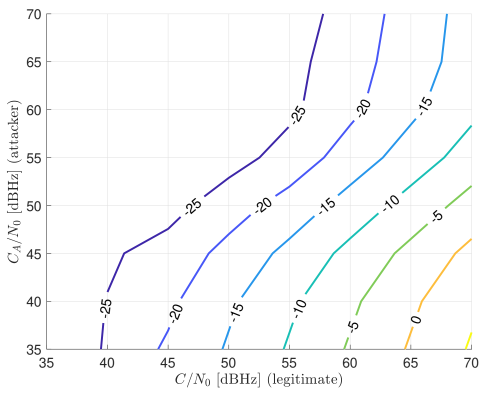


Figure 3.10: MSE $e(\hat{r}, t)$ computed assuming $C_A/N_0 = C/N_0 = 40$ dBHz averaged over one hour of signals.

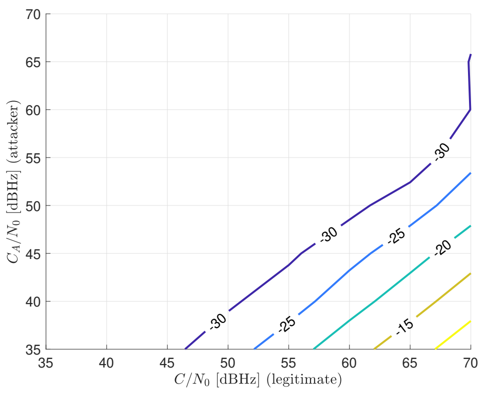
the attack strategy; however the attack strategy is in general not known by the victim receiver. This analysis points out that increasing the signal power, hence increasing the received C/N_0, allows achieving better detection performance. Indeed, increasing the signal-to-noise ratio allows the receiver to have a cleaner estimation of the received samples. On the other hand, increasing the received power also allows the attacker to improve its estimation of the unpredictable symbols. However, the results show that the latter effect does not fully compensate the former, so that increasing the received power is beneficial for anti-spoofing purposes at symbol level, as shown by Fig. 3.11.

3.6 Real world SCER attack

As discussed in this chapter, in the SCER attack the attacker acts as a matched filter for its estimation process, producing an estimate of the legitimate signal sample by sample. In the first version of our software package we modeled the estimation as an additive



(a)



(b)

Figure 3.11: $10\log_{10}(|E(\hat{r}) - E(r)|/E(r))$ as function of the receiver C/N_0 and C_A/N_0, averaged over one hour of signals. In (a) $E(\hat{r})$ is computed only in the first 50 μs of the symbols, while in (b) is computed over the whole symbol duration.

Gaussian noise, as described in [25]:

$$Z_l(n) = W_l + N_l(n), \qquad N_l(n) \sim \mathcal{N}(0, \sigma_Z^2(n))$$

with W_l representing the value of the l-th security code chip and Z_l the estimated sample value, with $\sigma_Z^2(n) = 2\sigma_s^2/n$. σ_s^2 is the variance of the noise samples at the spoofer's side and $n = 1, 2, \ldots$ the sample index.

In a more realistic attack environment the adversary needs to perform the estimation process on the legitimate signal through the use of a GNSS receiver. Only then the attacker can effectively generate the spoofing signal. We decided to build a more realistic implementation of the SCER attack as an added value to our software package. As both the necessary software components were already present, we combined them to provide a new attack mode we called *real world SCER*. To the best of our knowledge, an real-time implementation of the symbol-level SCER attack with a receiver-in-the-loop has never been proposed in the literature.

The above equation refers to the effect of a zero-delay attack, in which the attacker transmits it's forged signal perfectly synchronized with the legitimate signal he is receiving. As seen in Fig. 3.4, this will in general not be the case, as in order to impose an arbitrary position the attacker will have to anticipate some signals and delay the others. As in Sec. 3.3, $\Delta_i(t)$ indicates the difference between the geometric ranges of the spoofer and the falsified position with respect to a SV$_i$. The attacker can compute these values by using a constellation simulator such as the one integrated in our GNSS signal simulator.

We allowed the two pieces of software (signal simulator and receiver) to communicate and exchange data through TCP sockets, and design the real world SCER attack through the following steps:

1. the simulator produces the I/Q samples representing the legitimate signal received by the attacker (assumed to coincide with the victim in a *self spoofing* scenario). This signal is fed to the receiver, which is part of the attacker's equipment.

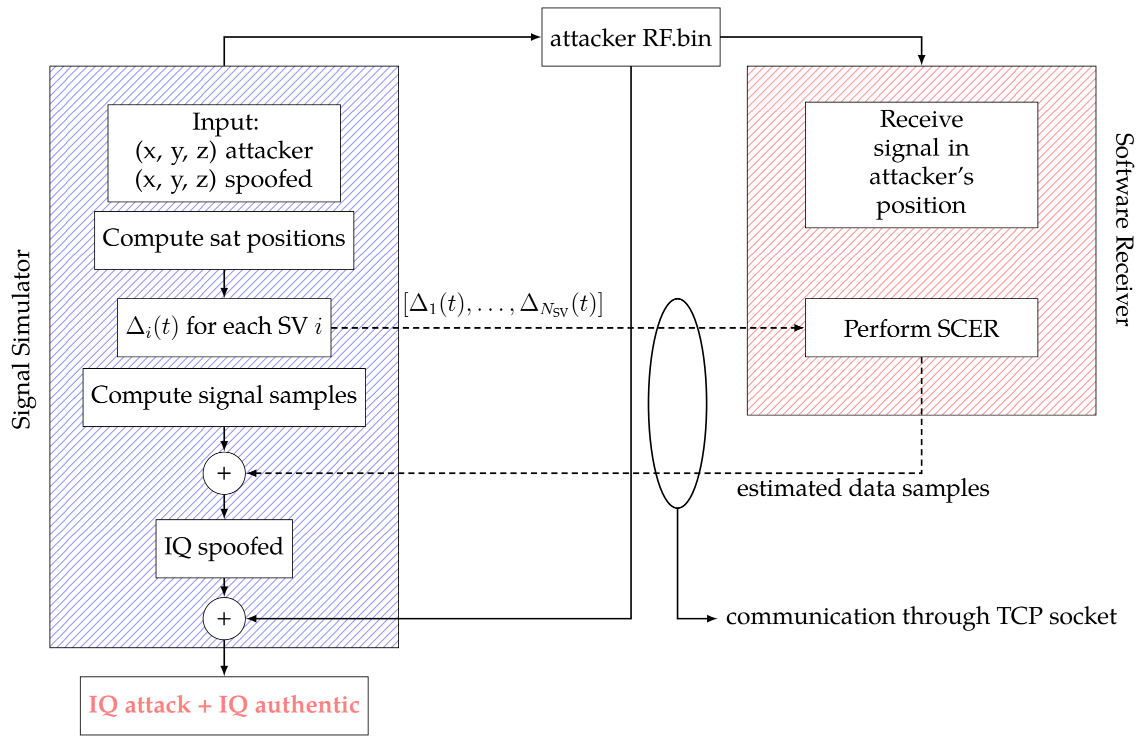


Figure 3.12: Block diagram of the real world SCER attack.

2. The simulator takes as input the spoofed trajectory and the attacker's trajectory (which coincides with the victim's one). It computes the pseudoranges and $\Delta_i(t)$ for each satellite and simulation step t.

3. At each simulation step t the simulator sends to the receiver all the values of $[\Delta_1(t), \dots, \Delta_{N_{SV}}(t)]$.

4. The following operations are performed in parallel:

   - The receiver performs the SCER estimation while anticipating or delaying the incoming legitimate signals of the time specified by $[\Delta_1(t), \dots, \Delta_{N_{SV}}(t)]$. The estimate is produced sample by sample and sent back to the simulator.
   - The simulator produces the spoofing signal, leaving the unpredictable bits blank while waiting for the estimated samples computed by the receiver.

5. The simulator receivers the estimated data samples and adds them to the spoofing signal, producing the attack output.

6. The attack output is finally added to the legitimate signal. The output can be fed to the receiver in order to assess its effects.

This real world SCER attack implementation was tested with positive results on the ublox M8T receiver against an automotive scenario in which the attacker is interested in simulating a deviation from the nominal route towards a toll-free road. The effect of the real world SCER attack has been found to follow theoretical results, thus validating our implementation.

3.7 Conclusions

This chapter investigated the feasibility of spoofing attacks against GNSS signals protected by NMA. The analysis was based on the most promising results in literature, which were combined to increase the success probability of the attack. The aim is that of evaluating the limitations and weaknesses of a state-of-the-art spoofing attack against NMA protected GNSS signals, in order to investigate what features can be leveraged for anti-spoofing purposes. A number of works in the literature suggested that NMA can be used to provide some level of ranging assurance. This work shows, both by analytical and experimental result, that the symbol unpredictability offered by the cryptographic functions in NMA does not offer significant improvement in range assurance.

The two components of this hybrid attack are:

- the Forward Estimation Attack, that works at the data layer, reducing the entropy of the unpredictable bits in the navigation message.
- the Security Code Estimation and Replay attack, that works at the signal layer and aims at producing a reliable estimate of the remaining unpredictable symbols, trading off symbol energy with time delay at the receiver's side.

The results show that even with a moderate C/N_0 the attacker can successfully generate a spoofing signal for the intended position. Indeed, the difference in the symbol error probability between the spoofing signal and the legitimate signal is an observable quantity only for low receiver's C/N_0 (e.g., $C/N_0 = 30$ dBHz), while for higher values (e.g., $C/N_0 = 40$ dBHz) the difference is in the order of 10^{-18}. The results obtained from the MSE analysis suggest that the legitimate signal may be distinguished from a spoofed one by looking at the difference in the MSE values corresponding to the first few samples. However, the uncertainty introduced by the SCER attack is visible only for extremely high values of the receiver's C/N_0 (e.g., ≥ 60 dBHz).

The above analysis pointed out that the observable effect of the considered spoofing attack belongs almost entirely to the signal layer. Hence, the protection offered by NMA at the data layer should be complemented by signal layer anti-spoofing techniques, working either at system level, [41,42], or at receiver level, [10,43]. In light of these considerations, NMA schemes should be designed to fulfill only the task of authentication and cryptographic integrity protection of the navigation message, while dedicated signal level anti-spoofing techniques should be designed to exploit NMA. The above analyses also serve the purpose of providing indications and guidelines for the the design of future signal components for GNSS signal authentication.

In a realistic self-spoofing scenario as the one reported in Fig. 3.4, it is likely that some of the satellites are in favor of the detection mechanism (i.e., the attacker needs to anticipate the signal). However, if the attacker employs multiple synchronized devices, properly placed in the receiver surroundings, each satellite can be simulated by the device that is placed in the most advantageous position, thus eliminating the need of anticipating signals. Another means of improving the attacker's advantage is that of introducing a time drift that would slowly force a delay in the receiver's clock, allowing the attacker to accumulate more samples before transmitting its estimate. A future extension of this work could combine the above techniques, thus designing a more sophisticated attack.

Chapter 4

Low-power selective Denial of Service on GNSS signals

The main focus in the previous chapters has been on spoofing strategies and spoofing detection and mitigation. Spoofing is regarded as a high integrity risk attack, since it potentially allows the adversary to inject forged PVT information inside the receiver module, undermining the security of underlying services. Spoofing is however considered a non-trivial attack, suggesting that even a less sophisticated adversary has means to cause damage. The attacker's target can be relaxed from position hijacking to service disruption, thus switching the focus to the category of attacks known as DoS attacks or jamming [44]. Traditional jammers achieve denial of service (DoS) by transmitting high power interfering signals, preventing the victim receiver to correctly track the genuine signal. The approach of a traditional jamming attack can be seen as brute-force: it disrupts the service over a certain area rather than selectively targeting a particular device or leveraging the known signal structure. Using such a generic strategy forces the attacker to transmit a considerably high power, making the attack easily detectable. Indeed one class of anti-jamming techniques is based on the detection and eviction of narrow-band interference signals in the frequency domain, where they have a sparse representation, as in [2]. As techniques operating in the transform domain can be demanding, others that work at the time domain have been explored in literature, as in [45], in which jamming mitigation is investigated through the use of adaptive notch filters. Pulse jammers, that are sparse in the time domain, can be mitigated through pulse blankers, detecting and excluding pulses whose power exceeds a certain threshold. A whole category of mitigation techniques operate in the spatial domain and are based on antenna arrays for determining the source of the interference and suppressing the undesired signals coming from that direction (e.g., beamforming and nullsteering) [46]. While jamming and mitigation techniques are still a hot topic, there is a whole unexplored middle ground between *brute-force* interference (jamming) and deceptive, more sophisticated attacks (spoofing) [47]. An example that lies in between is the *systematic jammer*, proposed in [27], which disciplines the jamming pulses to target the structure of GNSS data, making the attack more efficient and harder to detect. This approach drastically reduces the amount of power required for disrupting the PNT capability by preventing the victim from decoding the navigation message. By doing so, the attacker can even target a specific constellation, forcing the receiver to rely on the remaining ones.

In this chapter a novel jamming strategy is investigated that lies in the same *middle ground*, leveraging the structure of the GNSS signal to disrupt the positioning service more efficiently, by directly attacking the correlation process at the receiver. The proposed jammer aims at disrupting the lock indicators used by receiver, e.g., the code lock indicator (CLI) or the phase lock indicator (PLI): if the these metrics are degraded, the receiver is led to discard the signal.

The code lock indicator can be disrupted by a drop in the correlation value. Similarly to the so-called nulling attack, the attacker transmits a signal in the attempt of creating destructive interference with the legitimate one. The aim of the attacker is to degrade the received signal quality, rather than perfectly cancelling the legitimate signal, allowing to relax the constraints, both in terms of synchronization and of signal amplitude. On the other hand, a phase lock indicator monitors the correct tracking of the signal, which can be disrupted by adding an out-of-phase signal component.

The novelty of this approach, published in [48], lies in the minimization of the transmitted energy while allowing to choose the target constellation and even individual satellites. Moreover, the jamming and genuine signals combine destructively only for a specific receiver and geographical area, making it hard for distant interference monitoring stations to detect the attack due to its low power. This jamming signal is designed to have a higher probability of deceiving traditional jamming indicators and it is much harder to filter due to its similarity to the authentic signal itself.

4.1 System model and assumptions

Let us assume that the attacker (Eve) is located close to the victim receiver (Bob), in such a way that Eve's antenna is in line-of-sight (LOS) with respect to Bob's antenna. Eve can estimate the time of arrival of the authentic signals at Bob's side with a certain accuracy. Let us assume that this estimate is affected by mostly three error sources: the error in the victim's position, the error in the attacker's position and the time synchronization error. The combined effect of these error sources causes the victim's antenna to receive the authentic signal earlier or later than the estimated time, with ε representing this bias.

Beside the time alignment of the two signals, the attacker also needs to match the carrier of the legitimate signal (i.e., Doppler frequency and carrier phase). While it is possible to assume that the attacker knows the victim's motion, and can therefore reproduce the Doppler frequency with sufficient accuracy, matching the carrier phase is more challenging, as it requires centimeter level accuracy on the victim receiver's position.

Regardless of the specific GNSS constellation, the received signal is the superposition of several orthogonal PRNs modulated with data. This discussion relates to GPS L1 C/A, but can be trivially extended to other signals. Both the PRNs and the data are assumed to be publicly known, and the shortest time interval associated with a constant value modulated by the carrier is one chip period of the PRN sequence. Since PRNs are orthogonal, and they are assumed to be free of interference from one another, Eve can target each PRN individually. Under these reasonable assumptions, the aim is devising a low-power DoS attack by individually optimizing the jamming waveform for each target PRN. The resulting interference signal will be the superposition of all the designed waveforms.

4.2 Attack against the code lock indicator

Eve aims at disrupting Bob's tracking loop by transmitting an interfering signal. For the ease of derivation let us, for the time being, make the idealistic assumption from the point of view of the attacker that Eve knows the exact carrier phase and amplitude of the authentic signal in the victim's position. The aim of the jammer is to prevent the receiver from computing the PVT.

One common example of Code Lock Indicator (CLI) implementation can be found in [49]

$$\mu_{NP} = \frac{1}{N}\sum_{n=1}^{N} \frac{\left(\sum_{m=1}^{M} I_P\right)^2 + \left(\sum_{m=1}^{M} Q_P\right)^2}{\sum_{m=1}^{M}\left(I_P^2 + Q_P^2\right)} \tag{4.1}$$

where M and N are respectively the number of coherent and non-coherent integration intervals. Eve can disrupt the CLI by reducing the in-phase correlation power, I_P. Depending on the specific receiver implementation, if the CLI or the C/N_0 estimation are low enough, the receiver may end up dropping the signal even if tracking is still feasible.

Eve can then aim at reducing Bob's correlation as much as possible, while minimizing the energy of the jamming signal (in the statistic average). Assuming perfect PLL and DLL alignment, it is possible to write the GPS L1 C/A signal after carrier wipe-off as:

$$p(t) = \sum_{k=-\infty}^{\infty} c_k \operatorname{rect}\left(\frac{t-kT_c}{T_c}\right) \tag{4.2}$$

where $c_k \in \{-1, 1\}$ is the chip value and T_c is the chip duration.

Bob's metric for deciding the availability of the signal is the absolute value of the cross correlation between the signal and the local replica of the PRN. Let us assume the attacker can send his jamming waveform synchronized with the authentic signal with an error ε. Therefore, under Eve's attack, Bob's metric takes the form:

$$C_\varepsilon = \left|\int_0^{KT_c} p(t)y(t,\varepsilon)dt\right| = \left|\int_0^{KT_c} p(t)(s(t)+j(t-\varepsilon))dt\right| = \left|KT_c + \int_0^{KT_c} p(t)j(t-\varepsilon)dt\right| \tag{4.3}$$

where y is the superposition of the legitimate ($s(t)$) and jamming signal ($j(t)$), $p(t)$ is the local replica of the PRN, assumed to be perfectly synchronized with $s(t)$, and $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon)$ is the synchronization error of the jamming signal with respect to $s(t)$. Since we assume that the coherent correlation is performed over a single PRN period, K is the number of chips in a whole PRN (e.g., $K = 1023$ for GPS L1 C/A). The second term of the rightmost sum in (4.3) corresponds to the reduction in the correlation value due to the jamming attack.

In order to ease the formulation of the objective function for the waveform optimization, it is assumed that the PLL and DLL alignment are maintained during the attack, neglecting the effects of misalignments on the discrimination functions as well. However simplistic, this preliminary analysis solely aims at investigating low power approaches to jamming waveforms in order to achieve correlation loss with relatively contained effort.

4.2.1 Attack optimization

Eve's goal is to optimize the waveform $j(t)$, minimizing its energy subject to a constraint on the expected correlation loss ρ, where the expectation is taken with respect to the statistics of ε. This constraint is meaningful in our initial assumption that the attacker knows the carrier phase of the authentic signal, but it will be relaxed in later derivations, where this assumption will be dropped. ρ will then become a tuning parameter rather than an indicator of the correlation value. The optimal jamming signal can be then obtained by solving the optimization problem:

$$j^*(t) = \arg\min_{j(t)} \left\{ \int_{\tau=0}^{KT_c} j^2(\tau)d\tau \right\} \tag{4.4}$$

subject to

$$E\left[C_\varepsilon\right] \leq \rho K T_c \tag{4.5a}$$

$$0 < \rho < 1 \tag{4.5b}$$

Let us relax the constraint by exchanging the expectation operation and the absolute value, thus leading to:

$$\left| KT_c + E\left[\int_{t=0}^{KT_c} p(t)j(t-\varepsilon)dt \right] \right| \leq \rho K T_c \tag{4.6}$$

that can be rewritten as:

$$-(\rho+1)KT_c \leq \int_{-\infty}^{+\infty} f_\varepsilon(a) \int_0^{KT_c} j(t-a)p(t)\,dt\,da \leq (\rho-1)KT_c \tag{4.7}$$

where $f_\varepsilon(a) = \frac{1}{\sigma_\varepsilon\sqrt{2\pi}} e^{\frac{-a^2}{2\sigma_\varepsilon^2}}$. By performing a change of variable the order of the integrals can be switched, yielding:

$$b_1 \leq \int_0^{KT_c} j(t)q(t)dt \leq b_2 \tag{4.8}$$

where

$$q(t) = \int_{a=-\infty}^{+\infty} f_\varepsilon(a)\, p(t-a)da, \quad b_1 = (1-\rho)KT_c, \quad b_2 = (1+\rho)KT_c. \tag{4.9}$$

Note that using (4.2) we can rewrite $q(t)$ as function of the standard normal CDF function $\Phi(\cdot)$:

$$q(t) = \sum_{k=-\infty}^{+\infty} c_k \left[\Phi\left(\frac{kT_c/2}{\sigma_\varepsilon}\right) - \Phi\left(\frac{(k-1)T_c/2}{\sigma_\varepsilon}\right) \right]. \tag{4.10}$$

The problem is thus reduced to minimizing of the energy of $j(t)$ subject to a constraint on the inner product $\langle j, q \rangle$. Since no contribution to the inner product is given by the component of $j(t)$ that is orthogonal to $q(t)$, the optimal jamming waveform will have the form $j(t) = \beta q(t)$.

In deriving (4.6) from (4.5a) we relaxed the constraint and thus expanded the set of solutions. Therefore, the optimal solution obtained from (4.8) needs to be verified a-posteriori, to check whether it was also a solution of the original problem, or was introduced by the constraint relaxation. In the former case, the solution is also optimal for the original problem, whereas in the latter case a (possibly suboptimal) solution can be found by scaling the solution to the original constraint. Indeed, it is sufficient to project the waveform back into the set of legitimate solutions (i.e., a subspace of the space of solutions to the looser constraint (4.7)).

Each solution is characterized by ρ and σ_ε i.e., respectively the residual correlation factor and the distribution of the synchronization error. The solution to this problem provides Eve with the lowest energy waveform that allows to reduce the correlation value of the required amount, being robust to the timing errors of the attacker in the statistic average.

4.3 Attack against the phase lock indicator

In the previous section a jamming waveform was derived under the optimistic assumption that Eve can achieve perfect carrier phase synchronization with the authentic signal. Let us analyze a more realistic attack by dropping this assumption on the carrier phase alignment.

Beside the CLI, the receiver can usually rely on another lock indicator, the phase lock indicator (PLI), used in combination with the CLI for deciding whether to discard a signal if deemed as lost lock.

A common implementation of the PLI is:

$$\cos(2\Delta\varphi) = \frac{\left(\sum_{m=1}^{M} I_P\right)^2 - \left(\sum_{m=1}^{M} Q_P\right)^2}{\left(\sum_{m=1}^{M} I_P\right)^2 + \left(\sum_{m=1}^{M} Q_P\right)^2} \tag{4.11}$$

where $\Delta\varphi$ is the estimated carrier phase error. The signal is discarded when the PLI is below a certain threshold.

Intuitively, the PLI is close to 1 when the carrier alignment is perfect, and decreases as the quadrature component increases. A possible attack strategy therefore is to continuously rotate the carrier phase of the waveform obtained in the previous Section to make sure that at certain time the PLI decreases, either because the I_P component is reduced (when the jamming signal is summed with $\varphi \simeq 180°$) or because the Q_P is increased (when the jamming signal is summed with $\varphi \neq 0°$).

This can be achieved by a multiplying the optimal waveform $j^*(t)$ of (4.4) with a complex exponential, yielding the new jamming waveform

$$j'(t) = j^*(t)e^{i2\pi f_j t} \tag{4.12}$$

where f_j is an arbitrary frequency that controls the rotation speed of the waveform.

The impact of the attack heavily depends on the choice of f_j and on the victim receiver internal implementation, as discussed in the next section.

4.4 Numerical and experimental results

4.4.1 Simulation Results

The following results are derived from MATLAB simulations.

Fig. 4.1 represents the optimal waveform for a portion of PRN 1 of GPS L1 C/A. It can be noticed that as σ_ε decreases, the optimal waveform has lower amplitude and the shape of the jamming pulse becomes close to that of the PRN. This is due to Eve's sub-chip synchronization. On the contrary, for $\sigma_\varepsilon \geq 1$ the impulses spread over multiple chips, concentrating a greater amount of energy in batches wherein a value dominates the opposite one.

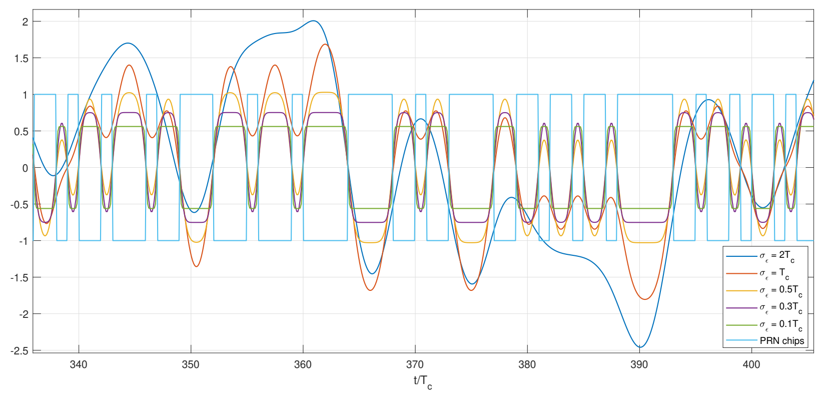


Figure 4.1: Optimal jamming waveform for different values of σ_ε and $\rho = 0.5$, in PRN 1 of the GPS L1 C/A signal.

Fig. 4.2 shows the value of the correlation between Bob's local replica $p(t)$ and the received signal $y(t)$. It can be noticed that for high values of σ_ε the correlation is significantly reduced even for wide relative shifts between the jammer and the signal. However, for $\sigma_\varepsilon \leq T_c$, a relative shift between $j(t)$ and $s(t)$ of a few chips is enough to restore the correlation value.

Even though Eve's synchronization is loose, i.e., σ_ε grows, there are always portions of the PRNs where the spreading code takes a value more frequently than the other. These portions will be targeted by Eve's jamming waveform, as shown in Fig. 4.3. It can be observed that for $\sigma_\varepsilon = 200T_c$, that corresponds to Eve having almost no information about synchronization, the optimal waveform degenerates to a CW amplitude modulated

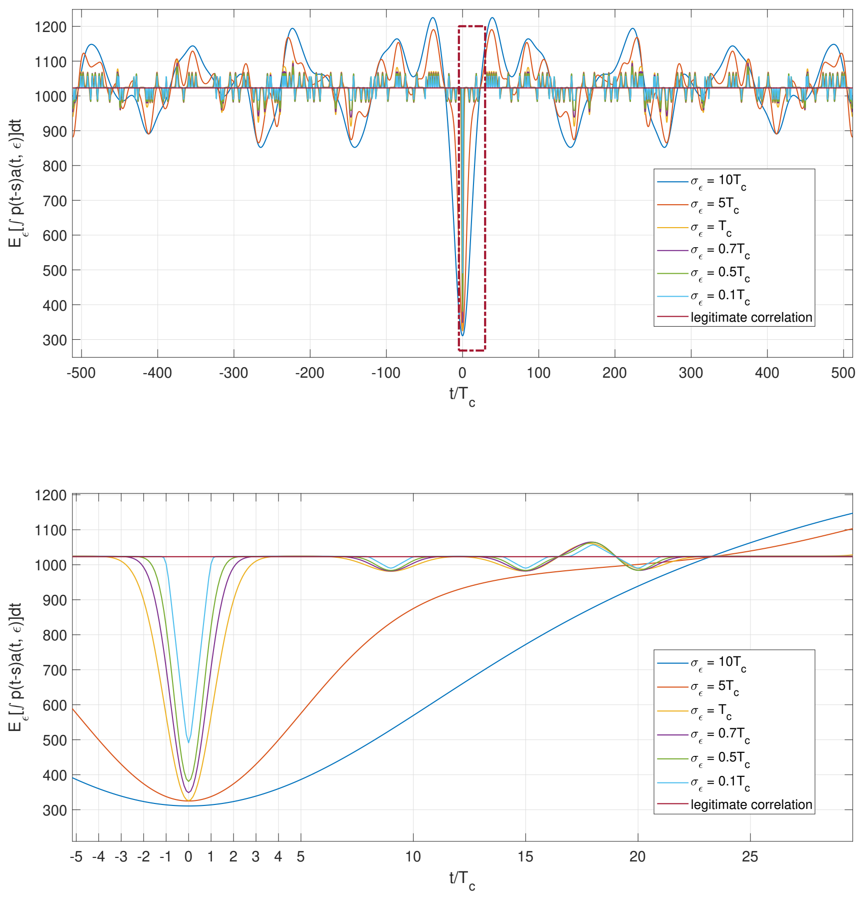


Figure 4.2: Plot of the expected correlation $E[C_\varepsilon]$ between $p(t)$ and $y(t)$ as a function of the synchronization error.

by a pseudo-sinusoidal signal with period the whole PRN.

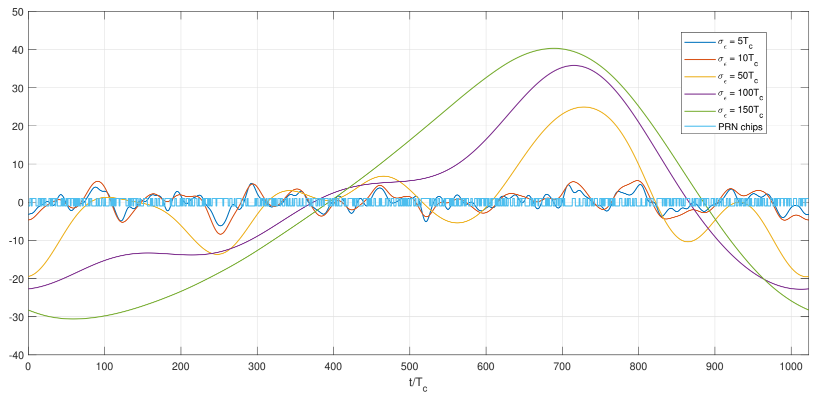


Figure 4.3: Degeneration of the optimal jamming waveform for high values of σ_ε and for $\rho = 0.5$.

Fig. 4.4 shows the power spectral density of the jamming waveform. It can be noticed that for small σ_ε the power spectral density it's close to that of the legitimate signal, making it harder to both detect and to filter the interference signal. On the other hand, when σ_ε increases, the power spectral density of the jamming waveform tends to concentrate in the smaller band, making it easier for the receiver to detect the presence of interference.

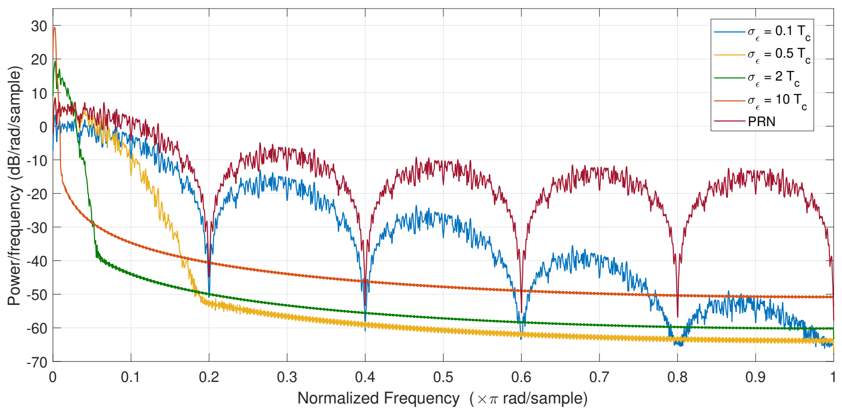
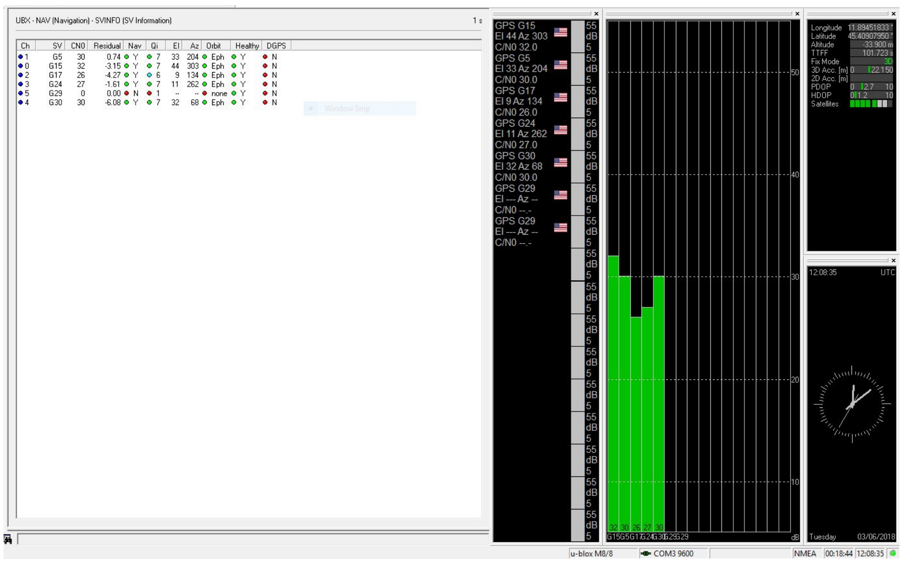


Figure 4.4: Welch power spectral density estimation for different jamming waveforms.

4.4.2 Experimental Results

The devised attack was implemented in the software signal simulator developed at the University of Padova. Specifically, the optimization problem was solved with quadratic programming in MATLAB, obtaining a sampled version of $j^*(t)$ for each PRN and superimposing the result to the authentic GNSS signal. The resulting waveform was modulated with a bladeRF SDR and fed to a u-blox M8T receiver that allows the assessment of correlation measurements. Fig. 4.5 and Fig. 4.6 show the experimental results relative to the attack against the CLI. Five GNSS signals were generated, each one corresponding to a different SV, with low C/N_0. After the u-blox receiver obtains a fix (Fig. 4.5), a jamming signal is added (Fig. 4.6) and the value of the C/N_0 estimation becomes 10 dB-Hz lower, while the fix is lost (red circle). A different synchronization error ε is applied to each PRN repetition in the jamming signal, where ε is drawn from a Gaussian distribution with $\sigma_\varepsilon = 0.1T_c$. This was done in order to evaluate the effects of the code misalignment with respect to the jamming waveform. Moreover, it can be noticed that the jamming detector does not indicate anomalies (green circle).



UBX - NAV (Navigation) - SVINFO (SV Information)

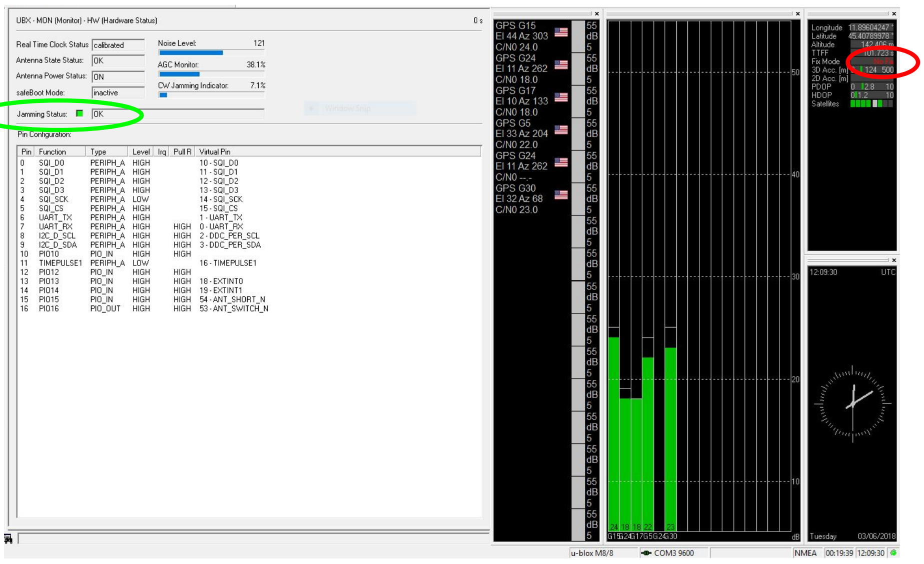
Ch	SV	CN0	Residual	Nav	Qi	El	Az	Orbit	Healthy	DGPS
1	G5	30	0.74	Y	7	33	204	Eph	Y	N
0	G15	32	-3.15	Y	7	44	303	Eph	Y	N
2	G17	26	-4.27	Y	6	9	134	Eph	Y	N
3	G24	27	-1.61	Y	7	11	262	Eph	Y	N
5	G29	0	0.00	N	1	--	--	none	Y	N
4	G30	30	-6.08	Y	7	32	68	Eph	Y	N

GPS G15
El 44 Az 303
C/N0 32.0
GPS G5
El 33 Az 204
C/N0 30.0
GPS G17
El 9 Az 134
C/N0 26.0
GPS G24
El 11 Az 262
C/N0 27.0
GPS G30
El 32 Az 68
C/N0 30.0
GPS G29
El --- Az --
C/N0 --.-
GPS G29
El --- Az --
C/N0 --.-

Longitude	11.89451833 °
Latitude	45.40907950 °
Altitude	-33.900 m
TTFF	101.723 s
Fix Mode	3D
3D Acc. [m]	0 22.150
2D Acc. [m]	
PDOP	0 2.7 10
HDOP	0 1.2 10
Satellites	

12:08:35 UTC
Tuesday 03/06/2018

Figure 4.5: Ublox M8T receiver before attack: nominal conditions.


UBX - MON (Monitor) - HW (Hardware Status) 0 s

Real Time Clock Status	calibrated
Antenna State Status:	OK
Antenna Power Status:	ON
safeBoot Mode:	inactive
Jamming Status: 🟩	OK

Noise Level:	121
AGC Monitor:	38.1%
CW Jamming Indicator:	7.1%

Pin Configuration:

Pin	Function	Type	Level	Irq	Pull R	Virtual Pin
0	SQI_D0	PERIPH_A	HIGH			10 - SQI_D0
1	SQI_D1	PERIPH_A	HIGH			11 - SQI_D1
2	SQI_D2	PERIPH_A	HIGH			12 - SQI_D2
3	SQI_D3	PERIPH_A	HIGH			13 - SQI_D3
4	SQI_SCK	PERIPH_A	LOW			14 - SQI_SCK
5	SQI_CS	PERIPH_A	HIGH			15 - SQI_CS
6	UART_TX	PERIPH_A	HIGH			1 - UART_TX
7	UART_RX	PERIPH_A	HIGH		HIGH	0 - UART_RX
8	I2C_D_SCL	PERIPH_A	HIGH		HIGH	2 - DDC_PER_SCL
9	I2C_D_SDA	PERIPH_A	HIGH		HIGH	3 - DDC_PER_SDA
10	PIO10	PIO_IN	HIGH		HIGH	
11	TIMEPULSE1	PERIPH_A	LOW			16 - TIMEPULSE1
12	PIO12	PIO_IN	HIGH		HIGH	
13	PIO13	PIO_IN	HIGH		HIGH	18 - EXTINT0
14	PIO14	PIO_IN	HIGH		HIGH	19 - EXTINT1
15	PIO15	PIO_IN	HIGH		HIGH	54 - ANT_SHORT_N
16	PIO16	PIO_OUT	HIGH		HIGH	53 - ANT_SWITCH_N

```
GPS G15
El 44 Az 303
C/N0 24.0
GPS G24
El 11 Az 262
C/N0 18.0
GPS G17
El 10 Az 133
C/N0 18.0
GPS G5
El 33 Az 204
C/N0 22.0
GPS G24
El 11 Az 262
C/N0 --.-
GPS G30
El 32 Az 68
C/N0 23.0
```

Longitude	11.89604247 °
Latitude	45.40789978 °
Altitude	142.406 m
TTFF	101.723 s
Fix Mode	No Fix
3D Acc. [m]	124 500
2D Acc. [m]	
PDOP	0 2.8 10
HDOP	0 1.2 10
Satellites	

12:09:30 UTC

Tuesday 03/06/2018

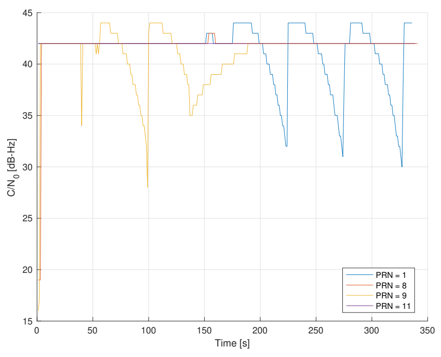
u-blox M8/8 COM3 9600 NMEA 00:19:39 12:09:30

Figure 4.6: Example of attack effects on a ublox M8T receiver.

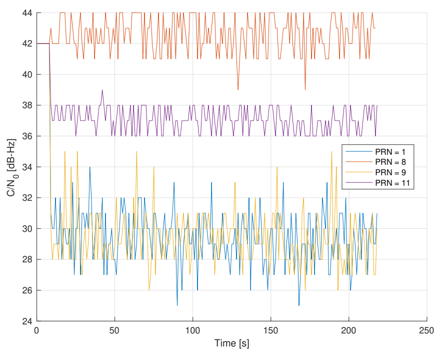
The results of the attack against the PLI are shown in Fig. 4.7, Fig. 4.8 and Fig. 4.9, that report respectively the C/N_0, the carrier phase and the pseudorange estimated by the receiver, for different jamming frequencies. The figures highlight that the behavior of the PLI is strongly influenced by the jammer frequency offset f_j. In Fig. 4.7a, Fig. 4.8a and Fig. 4.9a, the jammer frequency offset was set to $f_j = 100$Hz. Only some SVs were affected. Even if the effect on the C/N_0 estimator was moderate, the receiver lost the carrier phase lock several times and some spikes were registered in the pseudorange estimation in the corresponding instants. In Fig. 4.7b, Fig. 4.8b and Fig. 4.9b, the jammer frequency was set to $f_j = 1$ Hz. All SVs were affected and the entity of the impact on the C/N_0 estimation is due to a different initial error in the carrier phase alignment. The jammer signal indeed can be seen to interfere constructively with the legitimate signal of SV 8. The resulting C/N_0 estimation is thus higher than the actual value (42 dB-Hz). The opposite happens for SV 1 and SV 9, where the jammer signal interferes destructively with the legitimate one, causing an average correlation loss of more than 10 dB. A possible explanation for the stability of the estimated value is that $f_j = 1$ Hz is also the period of the C/N_0 estimation performed in the receiver. Therefore, the estimated C/N_0 is not affected by big fluctuations and remains almost constant around a certain value. Even if the C/N_0 is not dropping to levels were a loss of lock would occur, the receiver contentiously declares a loss of carrier lock and stops producing observables Fig. 4.8b. This effect is best observed for SV 8, for which the receiver keeps reporting a loss of lock even though the C/N_0 is over 40 dB-Hz.

In Fig. 4.7c, Fig. 4.8c the jammer offset frequency was set to $f_j = 0.1$ Hz. It can be noticed that the impact of jammer offset frequencies lower that 1 Hz is more predictable: a clear periodicity of 10 s can be recognized between fading events and carrier loss of lock events. As in the 1 Hz experiment, the pseudorange estimation was unaffected.

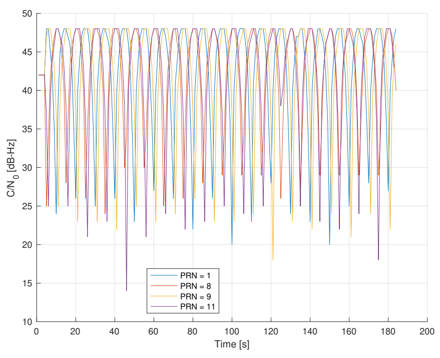
Some further investigation is needed to obtain an insight on how the jamming waveform truly affects the discriminators of the PLL and DLL. This analysis, which will be performed on the software receiver in order to have immediate control over the parameters and outputs, is left for future work.



(a)



(b)



(c)

Figure 4.7: C/N_0 estimated by the ublox M8T receiver. The jammer frequency is a) 100 Hz; b) 1 Hz, c) 0.1 Hz.

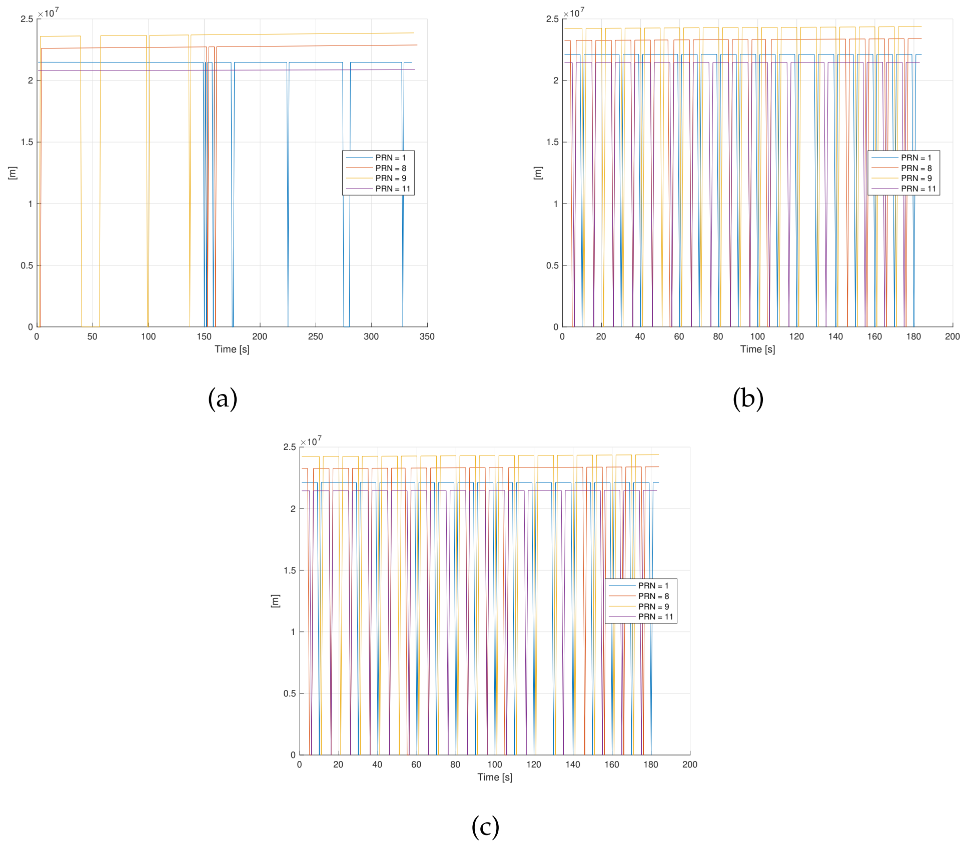


Figure 4.8: Pseudorange measurement of the ublox M8T receiver. The jammer frequency is a) 100 Hz; b) 1 Hz.

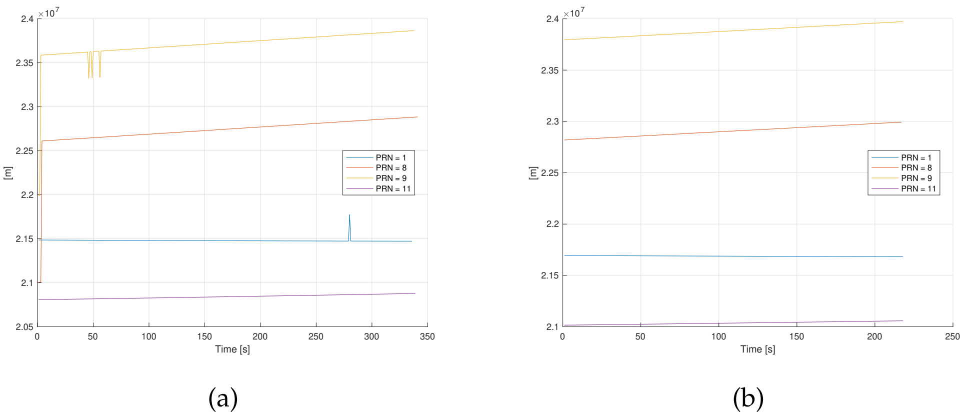


Figure 4.9: Pseudorange measurement of the ublox M8T receiver. The jammer frequency is a) 100 Hz; b) 1 Hz.

4.5 Conclusions

This analysis has investigated a novel class of DoS attacks against GNSS. Instead of using a brute force approach based on high transmitted power, a jamming waveform was devised that is matched to the spreading code of the target SV. The devised attack minimizes the transmitted power necessary to obtaining the target reduction in the correlation value, accounting for time synchronization errors between jamming and legitimate signals. The assumption on the attacker's phase synchronization was dropped by adding a frequency offset to the jamming signal, highlighting how a fictitious loss of lock can be triggered in a commercial grade receiver by targeting the PLL.

The devised jamming waveforms can be designed to have a power spectral density that is comparable to the legitimate signals, showing their effects after the correlation rather than at the frontend.

Chapter 5

Key management in GNSS applications

Cryptography is generally integrated in most communication system in order to fulfill security targets such as confidentiality, non-repudiation, integrity protection, and authentication. Key management comprises several functions ensuring that cryptographic material is securely generated, delivered, stored, updated and revoked, preventing unauthorized use of keys and attacks to their confidentiality and/or authenticity. Any cryptographic protocol (such as an authentication scheme) needs a security policy for dealing with both ordinary events (involving the regular operations of the system) and extraordinary ones (due to unexpected causes of any kind).

The initial service required in key management is that of establishing the crytpographic keys among the legitimate users, which starts with *key generation*, where cryptographic material is produced, ensuring that it is not easily predictable e.g., by using fresh randomness. For asymmetric mechanisms, private and public keys are generated by each entity for itself and public keys must then be delivered to other users through a channel that itself provides authentication and integrity protection. This operation is called *key distribution*, the second phase of key establishment. The distribution protocol shall respect practical requirements, defined by the specific application (e.g., time synchronization), in order to guarantee a transparent access to the service. It is good practice to periodically confirm or update all key material, e.g., by the Online Certificate Status Protocol (OCSP). The key management scheme shall ensure timely reception of key updates at the user side, thus preventing the unauthorized use of expired keys, which may lead to security vulnerabilities, or undermine the continuity of service. One cryptographic best practice is to protect the keys by minimizing their *cryptoperiod*, the time during which a key is used before a new one is issued. A shorter cryptoperiod limits the amount of information that is protected by the same key, the time available for cryptanalytic attacks, and the exposure time of the system in case of key compromise. Nevertheless, in bandwidth constrained scenarios such as GNSS, the frequency of key update operations has a critical impact on *communication overhead*, since the system needs to broadcast additional data in the form of key management messages. This trade-off between bandwidth and security is a critical driver in the choice of a cryptographic key management scheme, as security needs to be maximized while taking into account the limitations of the application environment.

Key establishment, update and revocation operations need to be assisted by mechanisms that allow to prove the legitimacy and authenticity of all involved message exchanges. These services can be provided by Public Key Infrastructure (PKI): a hierarchical organization of public keys with several layers. *Higher layers* shall be more robust (i.e., based on stronger cryptographic primitives with higher security parameters) in order to authenticate the messages for the management of *lower layers*, such as key updates and revocation. Lower layers can thus be designed with less stringent requirements, such as shorter signatures and public keys, allowing them to cope with performance targets and bandwidth constraints. The *chain of trust* induced by this architecture propagates the reliability of higher layers to lower ones, whose security is guaranteed by means of *public key certificates*. A certificate contains a public key and some information that binds it to a specific context, i.e., its version, serial number and validity period, and the issuer identity. All this information is signed by a Certificate Authority (CA), whose signature is included in the certificate as well.

The above approach establishes further layers of security whose task is to guarantee that all system maintenance operations are issued by the rightful authority. It is recommended to authenticate key management messages with a higher level of security with respect to the target system served by the infrastructure: the compromise of higher layer keys would indeed allow an attacker to take over the entire system, with no chance of recovery through rekeying. The result is a natural layering of the authentication keys, where higher layers authenticate lower ones [50]. This kind of structure is widely exploited in key management architectures over different types of network.

Key management systems may also offer additional services, such as a *group management* mechanism to take care of different user categories and a *user revocation* mechanism to allow the exclusion of subsets of users. The challenge is to design a key management scheme tailored to the GNSS scenario that is able to integrate multiple services within its structure, accounting for diverse needs and service requirements.

Overview of key management schemes employed in present networks

The Extensible Authentication Protocol (EAP) is a widely used standard [51] that allows to support various ciphersuits and authentication mechanisms at multiple layers. The EAP-AKA version, which stands for *Authentication and Key Agreement* [52], is compatible with UMTS and LTE networks, where it is used to achieve mutual authentication between users and network management entities [53–56]. This protocol starts with the negotiation of a master key: most of the other cryptographic keys needed for key management can be either directly derived from it or negotiated through a secure authenticated channel. AKA defines a set of request and response messages that must be exchanged between two parties in order to achieve mutual authentication. The challenge-response nature of this KM protocol is supported by the presence in most wireless communications of a duplex channel between users and network entities. The same does not hold for GNSS, where the channel from satellites to end-users is often the only one available, and user authentication is unnecessary. In LTE networks a successful mutual authentication allows the user to retrieve a 256- bit session master key (called ASME key) that allows the derivation of several other keys: encryption keys, integrity protection keys and authentication keys for communications with other network base stations (eNBs). LTE key management scheme thus exploits a hierarchical approach: a master secret key stored in the User Equipment allows the negotiation of the intermediate session master key (that can last from hours to a few days, depending on the receiver movements), from which several other keys are independently derived, to enable cryptographic key separation.

The WiMax standard for high rate transmissions through wireless networks supports encryption, integrity protection and user authentication [57]. Once again the focus of key management is mainly on access control, as support for mutual authentication was only introduced in later versions of the standard, to repair potential security flaws. This standard exploits the Private Key Management protocol for user authentication and key negotiation, which is composed of two phases. During phase one, the subscriber station sends to the base station its RSA public key certificate. The authenticator verifies all provided information and decides whether to grant or deny access to the network; in the former case, an encrypted symmetric authorization key is sent to the subscriber station. This scheme is well fitted to WiMax networks due to their high bandwidth and duplex features, as certificates carry a large amount of information (IDs, validity periods, signatures, etc.). Conversely, for the same reason, this kind of protocols do not fit GNSS restrained resources. Phase two has the aim of negotiating encryption and integrity protection keys to be used for secure data exchange. This negotiation proceeds with the derivation of a *key encryption key*, that is used to securely wrap and transmit cryptographic material, thus creating a multiple layer key hierarchy. In WiMax networks high bandwidth allows to introduce useful redundancy in security protocols: as an example, when key renewal needs to be triggered, the base station sends the encrypted version of both the old and the new keys, leaving less freedom to potential attackers (e.g., who might pretend to be the base station and distribute fake keys). This expedient is hardly applicable in the GNSS context, where it is critical to reduce the amount of exchanged information.

In WiFi networks too, key management starts with the establishment of a 256-bit master key, either manually or via EAP protocol [58]. A key hierarchy is then established through the derivation of pairwise keys (shared only between a station and the authentication server), and group keys. Similarly to the previously described schemes, keys with long cryptoperiods are used to derive temporary session keys, valid for a limited time and a specific context, such as the Pairwise Transient Key. A four-way handshake protocol is indeed exploited to establish this key, which in turn is used to derive lower layer keys for encryption and integrity protection. This approach, making use of key negotiation and derivation, is presently employed by most security protocols, such as the Transport Layer Security (TLS) protocol for secure end-to-end internet connection [59].

Broadcast television is an application scenario that is closer to GNSS in that bandwidth is a precious resource that gets mostly consumed by data [60]. Here, though, the main concern is user authentication and access control for avoiding unauthorized access to premium contents. A three or four-layer key hierarchy is exploited where shorter keys are updated really often: the keys that directly encrypt the contents might have a few-seconds cryptoperiod. Upper layer keys are used as group-keys to make communication more efficient and reduce bandwidth consumption. The highest level of the hierarchy consists in a master secret key that is pre-installed in the user smartcard.

This overview points out the crucial role of key management in every communication protocol that employs cryptographic mechanisms for providing security services. In the GNSS open service, the system primary concern is source authentication and integrity protection, while user authentication and access control are not required, nor applicable. The design of a suitable key management scheme for this scenario should thus depart from the above described approaches. However, the choice of appropriate algorithms and policies should be driven by the knowledge of strengths and weaknesses of well known key management schemes. Two ideas can be drawn from the above discussion: a key hierarchy allows to have more control over the system, by regulating the cryptoperiod of each key and separating their role, as good practice recommends. Moreover, although two-way protocols enhance the security of key management, they can not be implemented in GNSS due to its broadcast nature.

Section 5.1 tackles the design of a cryptographic key management scheme for GNSS data authentication, proposing an implementation tailored to the Open Service of Galileo. This proposal was published in [61].

Section 5.3 tackles the design of a key management scheme for access control in bandwidth constrained scenarios with stateless receivers (i.e., whose connection is not continuous in time). This work is an extension of [62] and is motivated by the Commercial Service of the Galileo GNSS constellation, which may offer additional services to a restricted set of users. An illustrative adaptation to the Galileo Commercial Service signal is proposed to evaluate the performance of the protocol in terms of communication overhead, which results to be reasonable for GNSS applications. The analysis and the proposed protocol are valid for any bandwidth constrained broadcast application scenarios.

5.1 Key management for GNSS open service data authentication

The GNSS navigation message is disseminated to the users at a low rate (e.g., 120 bps for Galileo E1B and 50 bps for GPS C/A and L1C), such that the modulation of the ranging signal by the message has a negligible impact on the precision of ranging measurement. The frequency of update of the navigation message is low, allowing adequate demodulation performance for users in a wide range of environments. In designing an efficient NMA scheme for GNSS, an optimal tradeoff must be sought among security, authentication performance, communication overhead, robustness (e.g., to channel errors and data loss), and receiver requirements.

GNSS is inherently asymmetric, i.e., one system broadcasting messages to many receivers, each receiver independently needing to verify the authenticity of the messages. However, traditional asymmetric authentication schemes based on digital signatures, such as DSA and its many variants (with elliptic curve arithmetic, Schnorr, etc.) are hardly applicable to this context. This is mainly due to the size of the public keys and signatures that are necessary to achieve the required level of security, which on such a bandwidth constrained dissemination channel has a significantly detrimental impact on authentication availability and performance. In addition, digital signatures tend to impose a high computational overhead on the receiver.

Several proposals for NMA in the literature [3–8, 24] have considered variations on broadcast authentication protocols such as Timed Efficient Stream Loss-Tolerant Authentication (TESLA). This protocol uses symmetric cryptography, minimizing the computational overhead of the receiver, and is flexible in that it can be configured to meet a range of requirements in terms of bandwidth and authentication performance. A distinctive characteristic of this protocol is the computation of a long chain of cryptographic keys, by repeated applications of a one-way hash function, from an initial randomly selected secret up to the final *root key*. The keys are then disclosed in opposite order with respect to their generation, so that the *root key* is disclosed first, and the initial key is last. Each disclosed key will be verified against an already verified key in the same chain and in turn used to verify a previously received message authentication code (MAC). The chain structure adds the benefit of a higher tolerance to packet losses, since even if the receiver misses one or more keys, later keys will allow to recover the missing part of the chain. This scheme bases its security on the fact that it is not computationally feasible to predict the value of future keys (yet to be disclosed), based on current and past keys, since the one way function cannot be efficiently reversed. The MAC provides data origin authentication as well as data integrity for all or part of the navigation message. Time synchronization is therefore critical, since together with authenticated time stamping of messages, MACs and keys, it ensures the receiver does not accept a message whose MAC may have been computed with an already disclosed key.

A digital signature is typically used to authenticate the root key in each chain and is broadcast as part of the authentication data.

An alternative solution based on the notion of *digital signature amortization* was proposed in [11], where a single signature is used to authenticate several messages in the same Issue Of Data (IOD), thus reducing the authentication overhead. This solution does not require time synchronization, and is therefore well suited also for discontinuous receivers. Moreover, the use of short chains, tailored to a single IOD period, reduces the success probability of collision based attacks, such as the one described [63]. As a downside, a more frequent use of the digital signature scheme is required, calling for a more frequent replacement of its private/public key pair.

A common feature of all the above mentioned NMA schemes is that their security ultimately relies on a digital signature. Therefore, it is essential to devise a mechanism that ensures the validity of the corresponding public keys.

5.2 Proposed Key management scheme for NMA

In order to enhance the protection of crypotographic key material for GNSS authentication, a hierarchical Public Key Infrastructure (PKI) can be adopted.

In the following, we will denote private and public keys for asymmetric cryptography as k and $\mathcal{K}$, respectively, while keys for symmetric mechanisms will be denoted as K. The proposed PKI is composed by three levels, that are:

level 1 a short-term private/public key pair $(k_{\text{L1}}, \mathcal{K}_{\text{L1}})$, used for the digital signature in the NMA mechanism.

level 2 a medium-term private/public key pair $(k_{\text{L2}}, \mathcal{K}_{\text{L2}})$, used to digitally sign security critical messages, such as changes (i.e., update/revocation) of the level 1 key pair, and possible system re-configurations. This key pair shall be more robust with respect to the ones at level 1.

root level a long-term private/public key pair $(k_{\text{CA}}, \mathcal{K}_{\text{CA}})$ bound to a Certificate Authority (CA), which authenticates both level 1 and level 2. The public key $\mathcal{K}_{\text{CA}}$ shall be pre-installed in the receiver memory. This layer provides the highest security level and thus the underlying signature algorithm shall be adequately strong.

With this structure, whenever the level 1 key pair needs to be renewed or revoked, an authenticated update message delivering the new level 1 public key shall be broadcast, along with its level 2 signature. Since bandwidth is a critical resource in GNSS scenarios, frequent transmissions of this kind may be too costly, either due to long level 2 signatures, long level 1 public keys or the combination of the two. In order to lighten the broadcast burden of this operation, without giving up the possibility of Over-The-Air Rekeying (OTAR), level 1 public keys can be preinstalled in the receiver memory. Storing a number of keys can indeed allow the users to be autonomous for a certain period of time. If the keys $\mathcal{K}_{\text{L1}}$ are stored in cleartext, this configuration leaves the system exposed to offline precomputation attacks. Indeed, an attacker could attempt to retrieve private keys from the corresponding public keys long before their coming into use, increasing his success probability. Even though such an attack will require time, high computational capabilities and memory, it is customary to assume the worst case scenario where the attacker has no memory constraints and state-of-the-art computational power. To avoid such threat, an encrypted version of level 1 public keys will be installed in the memory, using encryption keys K_{enc}. As an example, a table could be built that, for each key at level 1, contains:

- the key ID, in clear;
- $\mathcal{K}_{\text{L1}}$ itself, encrypted with K_{enc};
- the CA certificate of $\mathcal{K}_{\text{L1}}$, denoted as $\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L1}})$, encrypted with K_{enc} as well. The certificate includes a set of parameters (e.g., the key ID, the SV ID and its expiration time), and a signature that protects them and the associated key, computed with k_{CA}.

For a more effective limitation on an attacker's capabilities to manipulate the PVT computation and thus to enhance the system robustness, each satellite will be associated to a different $\mathcal{K}_{\text{L1}}$: therefore, the compromise of one key will not affect the security of messages coming from other SVs, achieving independence among satellites. Instead of storing one certificate for the $\mathcal{K}_{\text{L1}}$ of each satellite, a single CA certificate can be used to validate the concatenation of the whole batch of keys linked to the same ID: $(\mathcal{K}_{\text{L1}}^{1,\ell}, \mathcal{K}_{\text{L1}}^{2,\ell}, \mathcal{K}_{\text{L1}}^{3,\ell}, \ldots)$, where $\mathcal{K}_{\text{L1}}^{j,\ell}$ indicates the level 1 public key for satellite j of batch with ID ℓ. Analogously, instead of defining a different key for the encryption of each $\mathcal{K}_{\text{L1}}^{j,\ell}$, a single key K_{enc}^{ℓ} can be used to encrypt the batch $(\mathcal{K}_{\text{L1}}^{1,\ell}, \mathcal{K}_{\text{L1}}^{2,\ell}, \mathcal{K}_{\text{L1}}^{3,\ell}, \ldots)$. On the other hand, to comply with the hierarchical paradigm, $\mathcal{K}_{\text{L2}}$ can be unique for all satellites, since it is stronger and harder to compromise, enhancing bandwidth efficiency. This key is thus used to enable updates of the underlying level 1 keys. Additionally, $\mathcal{K}_{\text{L2}}$ along with its CA certificate $\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L2}})$ and $\mathcal{K}_{\text{CA}}$ will be stored in clear, assuming that retrieval of the corresponding private keys is much harder than for level 1 keys. The resulting cryptographic information saved in the receiver memory is shown in Table 5.1.

$\mathcal{K}_{\text{L2}}$	ID_1	$\mathcal{K}_{\text{L1}}^{1,1}$	$\mathcal{K}_{\text{L1}}^{2,1}$	$\cdots$	$\mathcal{K}_{\text{L1}}^{30,1}$	$\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L1}}^{1,1}, \mathcal{K}_{\text{L1}}^{2,1}, \mathcal{K}_{\text{L1}}^{3,1}, \ldots)$
$\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L2}})$	ID_2	$\mathcal{K}_{\text{L1}}^{1,2}$	$\mathcal{K}_{\text{L1}}^{2,2}$	$\cdots$	$\mathcal{K}_{\text{L1}}^{30,2}$	$\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L1}}^{1,2}, \mathcal{K}_{\text{L1}}^{2,2}, \mathcal{K}_{\text{L1}}^{3,2}, \ldots)$
$\mathcal{K}_{\text{CA}}$	ID_3	$\mathcal{K}_{\text{L1}}^{1,3}$	$\mathcal{K}_{\text{L1}}^{2,3}$	$\cdots$	$\mathcal{K}_{\text{L1}}^{30,3}$	$\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L1}}^{1,3}, \mathcal{K}_{\text{L1}}^{2,3}, \mathcal{K}_{\text{L1}}^{3,3}, \ldots)$
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
In clear		Encrypted				

Table 5.1: Stored key material at the receiver's side for the proposed key management architecture, divided between in-clear and encrypted content.

When the system imposes a level 1 public key change, either due to scheduled expiration or in case of revocation, it broadcasts the decryption key for the new batch, signed with k_{L2}. Once it is disclosed, the next level 1 public key batch will automatically substitute the former one. Each K_{enc}^{ℓ} is only known by the system, which decides its disclosure time according to the scheduling of level 1 rekeying. Notice that the only purpose of K_{enc} is to decrypt a public key and its certificate: once disclosed, it can not harm the system in any way. In light of these considerations, K_{enc} can be designed as the key of a symmetric encryption scheme, allowing to save bandwidth. After the demodulation and verification of K_{enc}^{ℓ}, the receiver has access to the updated level 1 public key batch ℓ, which can be verified against its decrypted CA certificate. This solution retains the security of the long CA signatures, while minimizing the amount of data to be broadcast, transmitting only level 2 signatures.

The level 2 public key shall be updated through an aiding channel, e.g., a network link. This allows to reduce the cryptoperiod of $\mathcal{K}_{\text{L1}}$ without forcing the receiver to frequently access the aiding channel. In order to ensure service continuity, key management information must be continuously broadcast, allowing receivers to get the needed information at any time during the key validity period. This continuous broadcast of level 2 signature also allows to address the problem of level 1 key revocation. Embedding in each level 2 signature a timestamp (e.g., Z-count or GST) will prevent it from being replayed. It will thus become harder for an attacker to intercept and discard the revocation message or substitute it with an out of date signature, luring the receiver into using expired or corrupted keys.

Fig. 5.1 shows the chain of trust. The CA public key, $\mathcal{K}_{\text{CA}}$, is used to verify both $\mathcal{K}_{\text{L1}}$ and $\mathcal{K}_{\text{L2}}$ through CA certificates. $\mathcal{K}_{\text{L1}}$ and its certificate are available only after the reception of the corresponding K_{enc}, in turn authenticated by the level 2 signature received via broadcast. The level 2 signature also authenticates the current level 1 key ID and a timestamp. The white blocks are stored in internal memory, while the gray blocks are broadcast by the SVs. The red dashed area corresponds to the encrypted storage area. It is worth noticing how this key management scheme offloads a significant part of the key management data from the broadcast channel to the internal storage.

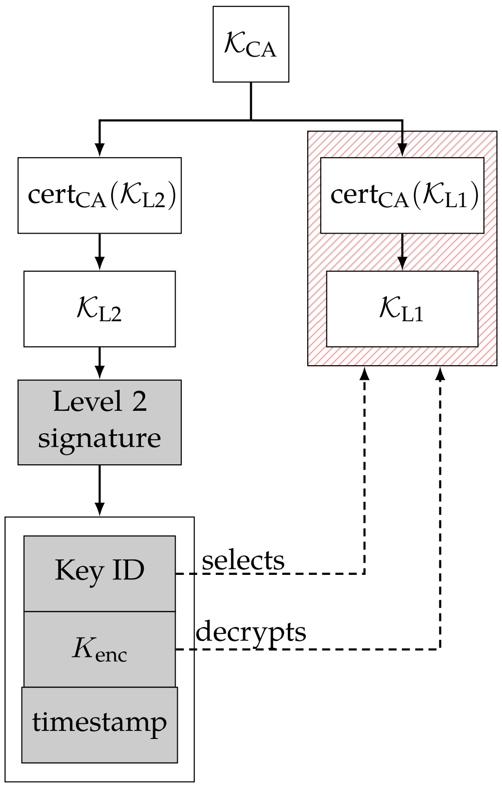

$\mathcal{K}_{\text{CA}}$
$\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L2}})$
$\text{cert}_{\text{CA}}(\mathcal{K}_{\text{L1}})$
$\mathcal{K}_{\text{L2}}$
$\mathcal{K}_{\text{L1}}$
Level 2 signature
Key ID
selects
K_{enc}
decrypts
timestamp

Figure 5.1: Chain of trust for the proposed key management architecture. The white blocks are stored in internal memory, while the gray blocks are broadcast by the SVs. The red dashed area corresponds to encrypted storage area. The $\mathcal{K}_{\text{CA}}$ is used to verify both $\mathcal{K}_{\text{L1}}$ and $\mathcal{K}_{\text{L2}}$. $\mathcal{K}_{\text{L1}}$ and its certificate are available only after the reception of the corresponding decryption key K_{enc} and its signature verification using $\mathcal{K}_{\text{L2}}$.

5.2.1 System Operations

The system generates a set of private-public key pairs, assigning an ID to each of them. Then a series of symmetric encryption keys are generated and used to encrypt level 1 public keys and certificates through a secure encryption scheme.

The system continuously broadcasts a set of data along with their level 2 signature. This data will be referred to as the Key Management Message (KMM), which is shown in Table 5.2. A KMM is composed by:

- a *level 2 signature timestamp*: a counter that avoids the replay of level 2 signatures. This could be for instance the system time corresponding to the first sub-frame where the first key management page of the current KMM is transmitted. The period of the KMM coincides with that of this timestamp; its duration is a system parameter and will be discussed in the next section.
- a *KMM counter*: a 1-bit counter that indexes KM messages. It is replicated in every KMM page in order to allow the receiver to discriminate to which KMM each page belongs. The KMM counter change is triggered by the increase of the level 2 signature timestamp, that is the most frequently updated field.
- a *configuration change timestamp*: this is either the time when last system configuration change has occurred or when the next scheduled reconfiguration will happen. The receiver can discriminate between the two cases by observing whether the timestemp refers to a past or future time. A system reconfiguration can be for instance the change of the digital signature scheme or the adoption of a different elliptic curve.
- the *key ID*: this is either the ID of the currently used $\mathcal{K}_{\mathrm{L1}}$ or the ID of next scheduled $\mathcal{K}_{\mathrm{L1}}$, according to the *key change timestamp*.
- the *decryption key*: the symmetric key that the receiver shall use to retrieve from storage the currently used level 1 public keys and the corresponding certificates.
- a *key change timestamp*: this is either the time when last $\mathcal{K}_{\mathrm{L1}}$ change has occurred or when the next scheduled rekeying will happen. The receiver can discriminate between the two cases by observing whether the timestemp refers to a past or future time. This timestamp marks the beginning of the cryptoperiod.

level 2 signature timestamp	KMM counter	configuration change timestamp	key ID	decryption key	key change timestamp	level 2 signature	total bits
32	1	32	6	128	32	512	743

Table 5.2: Example of Key Management Message (KMM), assuming the use of the GST as timestamp.

System reconfiguration

This mechanism is designed to notify receivers that a system reconfiguration has been scheduled. The actual information needed to reconfigure the system is not broadcast through the SIS, rather the receiver shall access the network and download the new configuration. This paradigm enhances bandwidth efficiency: although the event of a system reconfiguration is rarely supposed to happen, it requires the transmission of a considerable amount of data. Moreover, these data would have to be continuously retransmitted in order to support autonomous receivers, that can not rely on any aiding channel.

Level 1 rekeying

Two events might trigger an update of $\mathcal{K}_{\mathrm{L1}}$: the expiration of its cryptoperiod (scheduled update) or a key compromise (unexpected event). In the latter case NMA cannot be trusted, therefore an interesting option would be that of interrupting the nominal service in order to exploit also its bit allocation for fast level 1 rekeying. This solution would only be viable if the NMA service and the KM infrastructure are jointly designed. In this paper it is assumed that key management lies on top of NMA in a transparent fashion.

The system triggers a key change with the transmission of the new key ID, the corresponding decryption key, K_{enc}, and the key change timestamp. The level 1 key batch corresponding to the embedded ID will enter in service at the exact moment indicated by the timestamp. In case of scheduled rekeying, it might be convenient to disclose the whole update material in advance in order to ensure the continuity of service.

The advanced notification is needed to allow a seamless transition between the old and the new key. On the other hand autonomous receivers who do not possess the current $\mathcal{K}_{\mathrm{L1}}$ (e.g., devices who were inactive since last key change) will have to wait until the key change before being able to access the service. Thus, the advanced notification time should be dimensioned accordingly, in order to avoid excessive delays. It is worth noting that receivers who have access to the network might retrieve all the needed information through that channel. Moreover, the distribution of the decryption key exposes the corresponding public keys to cryptanalytic attacks; hence the level 1 keys shall be dimensioned to cope not only with the cryptoperiod but with the whole exposure time.

5.2.2 Receiver Operations

A receiver needs to have a set of pre-installed information. The CA public key shall be written in memory by the manufacturer. This is the root of the chain of trust and it is used for validating all underlying hierarchy. The remaining key material and system configuration can be either provided by the manufacturer as well, or securely bootstrapped via internet connection.

At cold start, before using the NMA service, a receiver must verify the validity of the information it has. Level 1 keys are verified against the level 2 signature, contained in the KMM. The authenticity of NMA can then be checked through level 1 signatures. It is good practice for the receiver to confirm the validity of the level 2 key and system configuration by periodically checking the network for scheduled renewals or recovery in case of key compromise.

During the normal use of the service, the receiver shall perform the authentication check from time to time, according to user requirements. If for any reason the receiver can not correctly demodulate the key management message, making it impossible to verify the level 2 signature for a certain amount of time, the underlying cryptographic material shall be declared invalid. If the authentication check is successful, the receiver shall additionally check the timestamp for inconsistency (e.g., out of order or replayed messages). After verifying the level 2 signature, the receiver shall check each field of the KMM. If the contained timestamps and key ID indicate an imminent key change, the device shall perform all the necessary operations for rekeying: the new key batch shall be deciphered and verified, and the key update time should be scheduled.

The proposed key management architecture brings the advantages of a hierarchical scheme to the GNSS context, enhancing the security of cryptographic material, while coping with bandwidth constraints. The resulting scheme is indeed robust to level 1 key compromise, allowing GNSS to recover from successful attacks directly through the SIS. Moreover, even if the level 1 scheme itself were broken, the system would still be able to securely alert autonomous receivers.

The proposed paradigm offloads information from the broadcast channel to the device storage. Since continuously transmitting the CA signatures can be prohibitive due to the limited capacity of the dissemination channel, the scheme exploits an intermediate-level signature instead, that is in turn authenticated by the CA. This signature is responsible for enabling $\mathcal{K}_{L1}$, which is directly validated by the CA (Fig. 5.1). The advantage in efficiency is even more tangible when the adopted level 1 signature scheme requires long public keys, such as [64] or [65]. Moreover, this architecture opens interesting trade-offs between signature length and cryptoperiod: by choosing a shorter cryptoperiod, the keys could be designed to have a lower security level, thus leading to shorter signatures, provided that the security level meets user requirements. Since changing the keys more often does not require additional transmission resources, reducing the key length up to the lowest accepted value can be balanced by a very short cryptoperiod. As an example, assuming an ECDSA signature, reducing the key length from 128 to 80 bit and the corresponding cryptoperiod from 6 to 1 months, the storage requirements would increase from 34 kbit to 116,5 kbit, respectively (assuming one year autonomy). This allows to spare 192 bits for each level 1 signature, and, assuming the transmission of one level 1 signature every 30 seconds, the equivalent reduction in bandwidth is of 6.4 bit/s.

The system is designed to cope with potential threats to layer 1 of the key hierarchy; in case layer 2 is compromised, either because a key is broken or because a vulnerability is discovered in the crypto-primitives of the scheme, no protection is guaranteed through the SIS. For this reason a robust level 2 signature should be selected and, as it was stressed before, its validity should be periodically assessed through the network.

Finally, it is worth observing that the presented scheme is flexible and modular. Indeed, it is possible to extend the hierarchical structure, increasing the number of layers, in order to build a more complex PKI and achieve higher resilience. A three-layer architecture has been proposed in order to cope with the bandwidth constraints of the

GNSS dissemination channel, but more layers might be added in case more resources can be dedicated to key management. Furthermore, this work has considered GNSS NMA as the target application, but the presented architecture can be adapted to any broadcast channel with limited data rate, or to any other cryptographic service such as GNSS signal layer authentication.

5.2.3 System parameters

In the phase of system design, several parameters can be set according to the mission requirements and use cases. In the following an explanation of these tradeoffs is reported, together with an example of system configurations that assumes the use of Galileo EDBS channel, i.e., 40-bit pages broadcast every 2 seconds.

- *Digital signature selection*: the three signature schemes and key sizes can be chosen independently. An example of how the level of security can scale from lower to upper layer is the following: layer one signature scheme should be selected autonomously by the service provider. It is good practice to set the minimum security level to 80 bits of security, as reported in [66]. Layer 2 should have a higher security level and thus it is recommended to employ a standardized signature scheme, since it must be harder to compromise. A candidate scheme can be ECDSA with 128 bits of security, which gives a signature size of 512 bits. It is worth noting that the literature can offer even more efficient schemes, e.g., [64, 65], although they are not standardized. Finally, for the CA signature scheme, a candidate can be 4096-bit RSA signature, as the size of the public key is not an issue for this key layer. However any equivalent security DSA or ECDSA signature scheme can be adopted as well.

- *Encryption scheme*: the scheme used for the encryption of the $\mathcal{K}_{\mathrm{L1}}$ batch and the corresponding certificates can be a secure symmetric cipher, e.g., AES.

- *Length of decryption key, K_{enc}*: there is a trade-off between security and bandwidth efficiency; in the current example a conservative approach has been preferred, selecting a 128-bit key. However, since the function of this key is that of avoiding pre-computation attacks on the public keys, choosing a shorter key (e.g., 80 bits) might still be feasible, depending on the user requirements.

- *Key ID*: $\mathcal{K}_{\mathrm{L1}}$ keys are indexed, and the system will cycle in order, incrementing the ID every time the batch expires. As an example, a 6-bit ID can be used to index a sequence of 64 keys. Allowing the index to change in a random fashion would increase the uncertainty of an attacker about the time scheduling of the keys, making it harder to target a certain key. On the other hand, if the key ID is progressively incremented by one, receivers at cold start can derive the exact number of key-updates they missed, and thus have a better knowledge of the system evolution. For instance, if the receiver has missed at least one update, and a new one is scheduled, he will not be able to access the service. If the keys are not ordered, the reason could either be that an update has been missed or that an attacker is targeting the receiver.

- *Frame allocation*: an example of bit allocation for the key management message consists in reserving one EDBS page for each subframe. This page can be organized as shown in Table 5.3, where 5 bits are used for the key management page sequence number, which is necessary for key management message reconstruction, allowing out-of-order reception. A 5-bit counter allows to index 32 pages of 34 bits each. One bit is reserved for the KMM counter, that allows the receiver to tell consecutive KM messages apart. Since in the current example, the KM message is 743-bit long, this results in 22 pages.

- *Scheduling*: as discussed above, a complete KMM can be transmitted in roughly one frame (i.e., 720 seconds). Since the KMM is the same for all SVs, it can be useful to exploit satellite diversity to improve the dissemination performance, similarly to what presented in [7]. A viable solution is to circularly shift the KMM pages, as in the example shown Table 5.4, where a set of four possible shifts was assumed. It can be seen how this proposal allows to reduce the minimum reception time with respect to the synchronized transmission of the same page sequence. In the represented situation 180 seconds are sufficient for the receiver to decode the whole KMM in case no transmission errors occur. The performance does not change depending on the switch-on time of the receiver. The number of shifts and the actual scheduling implementation can be optimized according to the specific constellation geometry.

- *KMM period*: this value is set by the update frequency of the level 2 signature timestamp. Updating the timestamp at each frame would offer optimal protection from data replay attacks, but the receiver will not be able to accumulate KMM pages over multiple frames. A trade-off shall be found between security and decoding performances.

- *Time gap for advanced notification*: this is the time offset with which the system will start transmitting information about the next $\mathcal{K}_{L1}$, in order to ensure service continuity with the desired probability. It can be optimized according to the target use case, and thus depending on the Cumulative Distribution Function (CDF) of the minimum time required for correct decoding of the KMM.

KM page ID	KMM counter	KMM
5 bit	1	34 bit

Table 5.3: Example of KMM bit allocation over a 40-bit page.

5.2.4 Performance analysis

This section aims at evaluating the performance of the scheme in terms of the time required to correctly receive the KMM. For the sake of tractability, we assume the channel to be stationary, where the transmission of each page is independent. Moreover, each satellite is considered to be independent of the others, and the actual constellation geometry is not taken into account. A complete analysis based on a more realistic channel model and specific satellite constellation is left for future work.

Time [s]	SV_1	SV_2	SV_3	SV_4
30	1	19	13	7
60	2	20	14	8
90	3	21	15	9
120	4	22	16	10
150	5	23	17	11
180	6	24	18	12
210	7	1	19	13
240	8	2	20	14
270	9	3	21	15
300	10	4	22	16
330	11	5	23	17
360	12	6	24	18
390	13	7	1	19
420	14	8	2	20
450	15	9	3	21
480	16	10	4	22
510	17	11	5	23
540	18	12	6	24
570	19	13	7	1
600	20	14	8	2
630	21	15	9	3
660	22	16	10	4
690	23	17	11	5
720	24	18	12	6

Table 5.4: Example of scheduling configuration, where four different circular shifts are defined. It is possible to notice that every 180 s window contains the whole KMM (assumed of 24 pages).

For the sake of simplicity a KMM of 24 pages instead of 22 is assumed, to align it with the frame duration.

Let's consider the transmission of a single page by one satellite. n is the number of transmissions required to correctly receive the page. A_i is referred to as the event: at transmissions i the page is correctly received. The probability of correctly receiving the page in no more than m page slots is given by:

$$\begin{aligned} P_{\text{page}}(n \leq m) &= P\left(\bigcup_{i=1}^{m} A_i\right) = 1 - \prod_{i=1}^{m} \left(1 - P\left(A_i\right)\right) \\ &= 1 - \text{PER}^m \end{aligned}$$

where PER is the page error rate. Let's now extend the computation to the case in which the KMM is composed by N_p pages. $A_{i,j}$ now becomes the event: at transmissions i page j is correctly received. Let's define $B_j = \bigcup_{i=1}^{m} A_{i,j}$ as the event: after m transmissions, page j is correct. The probability of correctly receiving the KMM in no more than M

page slots is:

$$P_{\text{KMM}}(n \leq M) = P\left(\bigcap_{j=1}^{N_p} B_j\right) = \prod_{j=1}^{N_p}(1 - \text{PER}^{m_j})$$

where m_j is the number of transmissions of page j,

$$M = \sum_{j=1}^{N_p} m_j$$

and the independence between pages is exploited.

The above derivation can be further extended to the case where N_{SV} satellites are in view.

$$P_{\text{KMM}}(n \leq M) = \prod_{j=1}^{N_p}\left(1 - \prod_{s=1}^{N_{\text{SV}}}(\text{PER}_s)^{m_{s,j}}\right)$$

where PER_s is the page error rate of satellite s, $m_{s,j}$ is the number of transmissions of page j by satellite s and

$$M = \sum_{j=1}^{N_p} m_j = \sum_{j=1}^{N_p}\sum_{s=1}^{N_{\text{SV}}} m_{s,j} \ .$$

For the absence of data relative to realistic scenarios, in the following it is assumed that the PER is equal for all channels:

$$P_{\text{KMM}}(n \leq M) = \prod_{j=1}^{N_p}(1 - \text{PER}^{m_j}) \ .$$

Fig. 5.2 represents the CDF of the decoding time as a function of the PER. It is possible to notice that if the average PER is below 10% the KMM is correctly retrieved in less than one E1B I/NAV frame (i.e., 12 minutes) with a high probability. In order to receive the KMM with the same probability when PER = 30% less than two E1B frames are necessary. If the PER grows up to 50%, the time needed to decode the KMM with high probability settles around 30 minutes.

Fig. 5.2 refers to the case represented in Table 5.4, where all the defined shifts are available, allowing optimal decoding performance. However, it is more realistic to assume that this is not always the case, as the satellites in view could in principle provide any combination of the devised shifts. The following scenarios are considered in the performance assessment, via CDF:

- *lower bound*: the satellites in view are transmitting the KMM in a synchronized fashion;
- *upper bound*: the satellites in view provide all the defined shifts of the KMM;
- *empirical*: a Monte Carlo simulation was run, randomly drawing satellite configurations, to average the above results in a more realistic fashion.

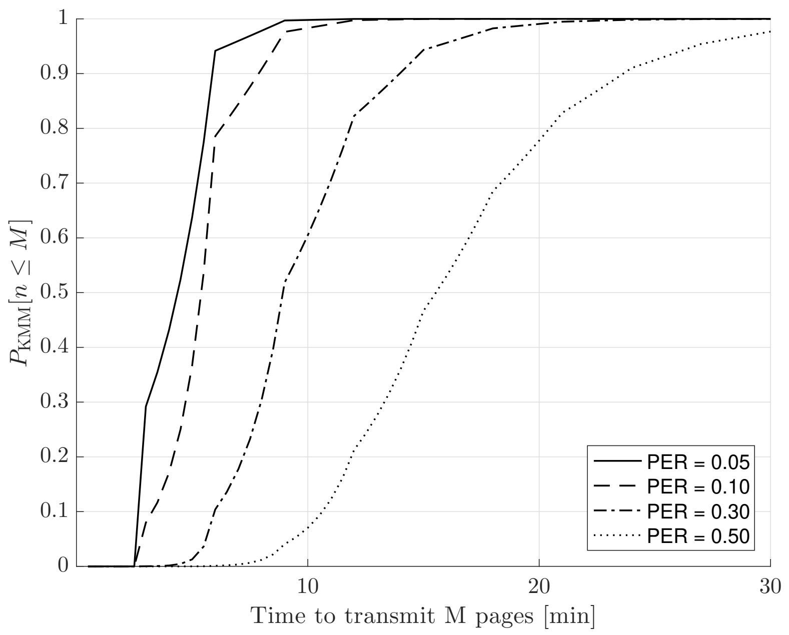
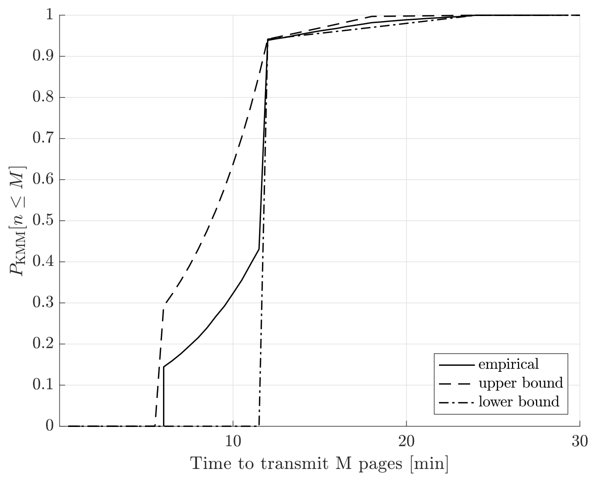
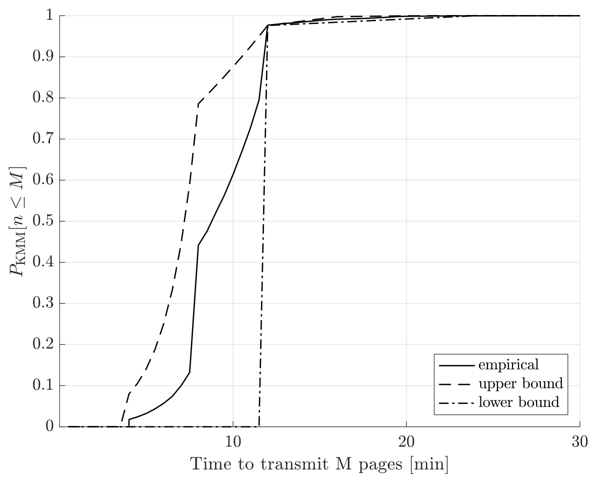


Figure 5.2: CDF of the decoding time for different PER values, in the case represented in Table 5.4. The time is computed assuming that a KMM page is transmitted every 30 seconds.

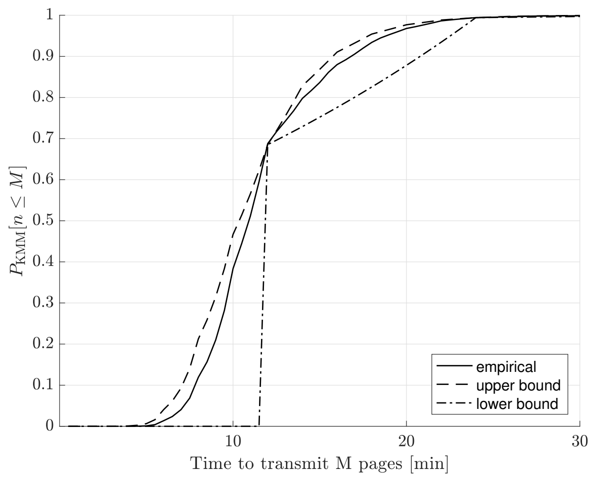
Fig. 5.3 shows that if the number of SVs increases, it is possible to correctly decode the KMM even in the case of high PER. In the three cases it has been assumed to exploit a scheduling scheme of two, three, and six circular shifts, respectively corresponding to the number of SVs in view. It is possible to notice how the use of an optimized scheduling (upper bound) scheme is expected to reduce the KMM decoding time with respect to a synchronized transmission (lower bound). As expected, the empirical curve lies in between the other two configurations. For two SVs, the empirical curve, after an initial advantage, approaches the upper bound. This happens because, due to the limited number of shifts, there is a 50% probability of having overlapping transmissions (Fig. 5.3a). With the increase of the defined shifts, the probability of having at least two non-synchronized SVs in view grows as well, leading to performance closer to the upper bound (Fig. 5.3c). It can also be noticed that in all the three cases the distinct curves coincide in correspondence to the end of a frame. This is due to the fact that, irrespectively to the page transmission order, at these time instants the amount of page-replica accumulated is the same; and having assumed that the errors are independent and identically distributed (i.i.d.), the probability of correctly decoding the KMM is the same.



(a)



(b)



(c)

Figure 5.3: CDF of the decoding time for different numbers of satellites and PER values. In a) 2 SVs are in view with PER $= 0.05$; in b) 3 SVs are in view with PER $= 0.1$; in c) 6 SVs are in view with PER $= 0.5$. The time is computed assuming that a KMM page is transmitted every 30 seconds.

5.3 Key Management for Access Control to Broadcast Services

The concept of selectively restricting the access to a resource is known in the literature as Access Control, and is often applied in several telecommunication scenarios through the means of cryptography. Encryption allows to protect the resource in such a way that it can be accessed only through the use of a secret piece of information (i.e., a cryptographic key), that may be private or shared among several users.

Broadcast communications are a particularly challenging field for the application of Access Control schemes, as by their own nature the same resource is distributed from a single source to many users, only a subset of which are entitled to it. While it is reasonable to assume that most of the legitimate users will cooperate with the central authority by protecting the secrecy of their access token, an access control scheme should account for the presence of malicious users (traitors), that will leak their own access information to non-entitled users.

The problem of access control in broadcast transmissions is known in the literature as *broadcast encryption* and it has been well investigated and applied to various scenarios. There are several examples of broadcast services that require an access control scheme.

- *Broadcast television* offers the access to programs and channels to subscribers only, encrypting the service and providing the access keys in a tamper resistant smartcard.
- In the scope of the *Internet of Things (IoT)*, some sensor networks require a central node to securely exchange broadcast messages with distributed sensors. In order to protect the network from malicious nodes access control is performed through the means of broadcast encryption.
- *Copyright* can also be considered a special case of broadcast encryption, where a stored content needs to be accessible only to authorized users.

The aim of the analysis carried out in [62] is to devise a key management scheme for access control in broadcast environments, characterized by the following features:

- the data rate is dramatically limited (e.g., tens or hundreds of bits).
- some of the users are stateless: i.e., they are not aware of all the past broadcast messages, as they are not constantly connected to the network.
- user revocation capabilities shall be supported to the extent allowed by the low data rate.

The target application is GNSS premium services. Galileo, the European GNSS, has envisioned a *Commercial Service* signal that will offer added value services with respect to the Open Service (OS). As reported in [67] the data rate is 500 bps and two additional signal components will facilitate advanced functions such as the integration of Galileo positioning applications with wireless communications networks, high accuracy positioning and indoor navigation.

The provided results, however, do not depend on the specific type of service, as this does not affect the overlying key management scheme: the only aspect that influences the design is the amount of exchanged information per unit time, and the rate of change of the authorized users set. So, even though GNSS services are the target application, the proposed key management scheme for access control is general and applicable to any access control scenario where the most critical resource is bandwidth.

The main contribution of this work is the extension and adaptation of the protocol defined in [62]. Some changes in the implementation of the scheme have been applied due to practical reasons and a theoretical security analyses has been added, together with a meaningful use case and possible application to Galileo's commercial service.

5.3.1 General Model

The general broadcast encryption scenario consists of a set $\mathcal{S}$ of possible users and a central site that broadcasts a message M encrypted with a key K_G to a dynamically changing *privileged set* $\mathcal{G} \in \mathcal{S}$ of authorized users, in such a way that non-members i.e., those in $\mathcal{S} \setminus \mathcal{G}$ cannot learn M (group confidentiality), as described in [68].

The main aspects that guide the choice of an appropriate broadcast encryption scheme are *confidentiality* (in the form of forward and backward secrecy and collusion resistance), *broadcast message length*, *scalability*, *unplanned eviction capabilities* and *secret storage requirements*.

- *Confidentiality* is a security service that allows to restrict the access to the target information to intended users, keeping it secret with respect to $\bar{\mathcal{G}}^t$. Confidentiality can be divided into two aspects that are less stringent: *forward secrecy* and *backward secrecy*. *Forward secrecy* is ensured if $\forall\ u \in \mathcal{G}^{t-1} \setminus \mathcal{G}^t$, even though u knows the previous group key K_G^{t-1}, it cannot find out the next ones, $K_G^t, K_G^{t+1}, \ldots$

  Backward secrecy is ensured if $\forall\ u \in \mathcal{G}^t \setminus \mathcal{G}^{t-1}$, even though u knows the current group key K_G^t, it cannot find out the previous ones, $K_G^{t-1}, K_G^{t-2}, \ldots$

  Collusion resistance is a security service related to confidentiality with respect to collusion of users that own outdated access material. In this paper we target confidentiality only *with respect to forward secrecy*, that is $\forall\ \mathcal{C}^t \subseteq \mathcal{G}^{t-1} \setminus \mathcal{G}^t$, users in $\mathcal{C}^t$ cannot retrieve the key K_G^t, not even by combining their secret tokens from previous authorization (whose validity has expired before time t).

- Scalability is related to the long term evolution of the system in two different aspects:
  - the information exchanged through the broadcast channel should scale at least sub-linearly with respect to $|\mathcal{G}^{t-1} \setminus \mathcal{G}^t|$ in order to maintain reasonable decoding performance.
  - the structure of the cryptographic scheme shall be elastic enough to cope with the departure of expired users and with the arrival of new ones: old private tokens will expire and new ones will be needed.

- The broadcast message length characterizes the bandwidth consumption. This value may vary according to the state of the system, e.g., number of authorized users i.e., $|\mathcal{G}^t|$, or membership changes, i.e., $\sum_{i=2,\ldots,t} |\mathcal{G}^{i-1} \setminus \mathcal{G}^i|$.

- Unplanned eviction capability refers to the possibility for the system to perform the unplanned revocation of any random subset of users, possibly multiple times.
- Secret storage requirement is the amount of secure (e.g., tamper resistant) memory that is needed by each user to store all its private cryptographic material necessary for accessing the protected resources.

5.3.2 General Features and Requirements

In the following we define a model for the underlying service, that is the system that will be protected by the key management scheme.

Three kinds of events are possible in the target system that modify the *privileged set* $\mathcal{G}^t$, that is the set of authorized owners of the group key:

- *join event*: a new user joins the privileged set;
- *leave event*:
  - **expiration event**: a user's authorization naturally expires and the user leaves the privileged set;
  - **unplanned eviction event**: a user's authorization is revoked by the system before the intended expiration date and the user is evicted from the privileged set.

The following paragraphs list some considerations that are particularly meaningful in the considered application scenario.

User's permanence time

Let us make the assumption that the intended permanence time of each user in the system is known by the central authority at the moment of it joining the privileged set, as it is reasonable e.g., for most subscription based services. The **knowledge on the permanence time** of users is valuable information, as it can be exploited to differentiate between the leave events due to authorization expiration and unplanned user eviction. The former case is supposedly known in advance and can therefore be planned, while the latter is impossible to predict. This distinction is suggested in [69] and [70] in the scope of multicast encryption for stateless receivers. These schemes are based on hash chains and binary hash trees, that are built by iterating one way functions and exploiting their properties of expansion and pre-image resistance.

Scalability and key structure

The number and time of join and leave events influences the requirements of the system for what concerns the size of the underlying structure. Most of the schemes in literature indeed rely on a specific data structure for organizing user and group keys (e.g., hash chains, binary hash trees, hierarchies, etc.), which need to be dimensioned before-hand in the scheme design phase. On one hand the size of such structures should be large enough to accommodate all the future changes in the privileged set, but on the other hand the communication overhead and storage requirements usually grow with this parameter. One solution is to periodically update the expired keying material and reassign it to new users. The new material however needs to be known to users in $\mathcal{G}^t$ but not to those in $\bigcup_{s=1}^{t-1} \mathcal{G}^{t-s} \setminus \mathcal{G}^t$. Improving the **scalability** of the scheme is therefore likely to impact the other performance measures.

Security

The paradigm of broadcast encryption deals with a group of users sharing a secret. This concept implies either a reciprocal trust among the members of the group or the existence of a mechanism to prevent members from leaking the shared secret. Since the former hypothesis is too optimistic, the existence of a *tamper resistant device* is a necessary assumption for a broadcast encryption scheme. One may argue that collusion resistance is irrelevant: if the secret is stored in an inaccessible memory indeed it is forever safe; on the contrary if it can be accessed by the user then the system should rather worry about the current users directly leaking the secret than former users trying to reconstruct it from their past information. However, in the long run the number of former users will be relevantly greater than the cardinality of the privileged set and collusion resistance grants the system a better control: the central authority shall worry only about the behavior of the current users. Moreover, even though a user in $\mathcal{G}$ leaks secret information, collusion resistance allows to restore confidentiality through a user eviction operation, provided that the system is capable of identifying the traitor.

Identification set message

According to [68] the communication overhead is composed of two contributions: the *broadcast encryption transmission* or *ciphertext* and the *identification set* message, used to identify the privileged set. The set identification message is necessary in case of unplanned eviction, since confidentiality requires that the new group key shall depend on the identities of the revoked users, which can not be known a priori (differently from expiration events). Most of the schemes in literature do not account for this overhead; however in low bandwidth applications such as GNSS, even the slightest increase in the amount of broadcast information may dramatically increase the transmission time, possibly impairing the usability of the system.

The identification set message is required for the identification of k users in a set of n, and therefore it has entropy $H(\mathbf{v}) = \log_2 \binom{n}{k}$. The length of this message increases with n and k, therefore for big systems that need the capability to perform unplanned user eviction many times, the identification set overhead will become even bigger than the key management message itself, as showed in figure 5.4. For these reasons it may be more appropriate for the system to divide users into groups. In case a specific user needs to be revoked, the system will have to revoke the whole group, impacting the other members, that will need to get a new authorized key through an aiding channel.

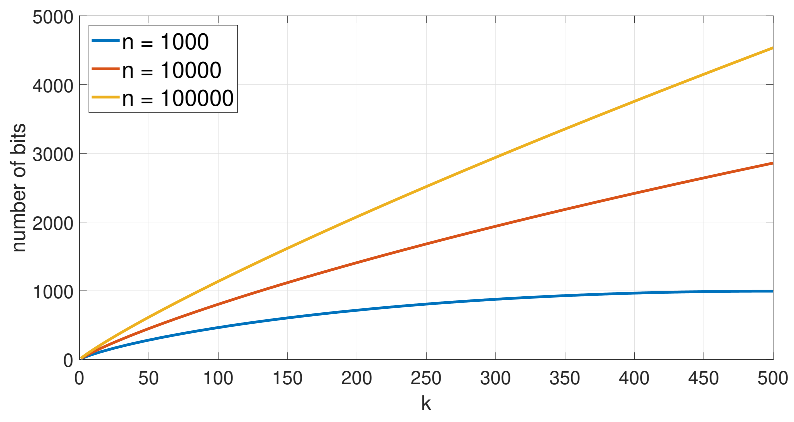


Figure 5.4: The number of bits needed to identify k out of n users in the system.

5.3.3 Related Work

The literature on broadcast encryption ensembles several solutions that focus either on reducing the bandwidth consumption or the storage requirements, with the aim of finding an optimal working point according to different application scenarios. In this paragraph a few schemes from the literature will be reported, that are worth taking into account in the design of an ad-hoc scheme for GNSS. An extensive literature review of broadcast encryption schemes can be found in [62].

The first broadcast encryption schemes have been proposed by Fiat and Naor in [68]. They start from non-collusion resistant schemes and extend them by exploiting a **combinatorial distribution of keys**, obtaining r-resilience (resilience to collusions of at most r users) and guaranteeing support for stateless receivers. The proposed protocol has $\mathcal{O}\big(r^2 \log^2 r \log n\big)$ communication overhead and $\mathcal{O}(r \log r \log n)$ secret storage requirement at the receiver side with r the maximum number of users outside the privileged set ($|\mathcal{G}^{t-1} \setminus \mathcal{G}^t| \leq r \quad \forall t$).

In [71] asymmetric encryption is exploited, combined with Shamir's polynomial based threshold secret sharing. The presented asymmetric key scheme allows the revocation of up to r users with a communication overhead of $\mathcal{O}(r)$. This systems also has *traitor tracing* capabilities: given a pirate device and a subset of suspected users, a confirmation test can be run in order to establish whether any traitor belongs to that subset.

Not all schemes in literature pursue all of the desirable features listed so far. The authors of [14] settle for a lower level of security (1-resilience) in order to gain on the other performance measures and simplify the structure of the scheme. Similarly, in [72] confidentiality requirements are relaxed, admitting a number of "free riders" inside the privileged set in order to overcome the performance bounds. [73] and [74] start from 1-resilient protocols and combine them with collusion resistant schemes in order to trade off communication overhead for the desired degree of collusion resistance. The scheme proposed in [16] for GNSS related applications gives up unplanned eviction capabilities in order to eliminate the communication overhead at $\mathcal{O}(1)$ secret storage requirements. In this scheme a number of hash chains at different hierarchies allow to have different user categories. Similarly, in [69] the MARKS scheme has similar features and exploits hash trees in an analogous fashion.

In [75] and [76] two schemes are independently designed based on the concept of **logic key hierarchy** (LKH), where users are organized as the leaves of a binary hash tree. However, this approach supports stateful receivers only. Various adaptations of LKH schemes to pay-TV systems have been proposed in literature, such as [77] and [78]. An extension of LKH schemes to a stateless version, namely "subset difference", is presented in [12]: the revocation of k users requires $\mathcal{O}(k)$ communication overhead and $\mathcal{O}(\log^2 n)$ secret storage. This scheme is one of the strongest for stateless receivers in literature as it provides perfect collusion resistance at reasonable performances. None of the above schemes reaches a communication overhead performance below $\mathcal{O}(k)$, with k the total number or revoked users since the start of the protocol.

The scheme selected in [62] for adaptation to the GNSS scenario is that presented in [15]: an asymmetric key broadcast encryption system based on the bilinear Diffie Hellman exponent assumption (BDHE) and bilinear maps. The scheme is collusion resistant and supports stateless receivers, achieving $\mathcal{O}(1)$ communication overhead and private key size, with a public key of size $\mathcal{O}(n)$. Even though the public key is linear in n, which could lead to prohibitive requirements for the memory of a smart card, it is *public*, therefore users could easily store this information in non-secure memory.

From the literature, three useful concepts can be drawn and exploited in the scheme design:

- a **time ordering of receivers**, according to the expiration of their authorization, allows to distinguish between leave events due to authorization expiration and unplanned eviction. The former can be planned and carried out without requiring any identification set message ([69,70]).
- **hash chains** have two useful properties: they allow to derive multiple keys from the same seed and they are computationally hard to invert. If the keys obtained through a hash chain are released starting from the end, one at a time, each of them can be verified by applying the hash function to the previous one. Moreover, any intermediate key in the chain allows to compute all the following (future) keys, which is convenient in the key distribution phase, where users can be provided with an intermediate key according to the number of keys they are allowed to have.
- organizing the keys in a hierarchical structure guarantees cryptographic separation between them, protecting each layer against attacks to weaker layers.

5.3.4 Explicit Formulation

In the following the broadcast encryption scheme based on bilinear maps (BMBE) from [15] is briefly described, and an adaptation to the scenario of interest is devised, fitting it to the requirements outlined in the previous sections. In [15] three different version of the scheme are proposed, where the secret storage requirements are balanced off with the communication overhead. The most interesting version of the proposal for the scope of this work is the one with $\mathcal{O}(1)$ communication overhead, $\mathcal{O}(1)$ secret storage requirements and $\mathcal{O}(n)$ non-secure storage requirements.

The BDHE assumption states that given a group $\mathbb{G}$ of prime order p, $\alpha \in \mathbb{Z}_p$ and the vector

$$(h, g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^\ell}, g^{\alpha^{\ell+2}}, \ldots, g^{\alpha^{2\ell}}) \in \mathbb{G}^{2\ell+1}$$

where g and h are generators of the group, it is hard to find $e(g,h)^{\alpha^{\ell+1}}$, where $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$ is a bilinear map.

System setup

Let $\mathbb{G}$ be a bilinear group of prime order p. The central station picks a generator g, $\alpha \in \mathbb{Z}_p$ and a random $\gamma \in \mathbb{Z}_p$. It computes $g_i = g^{\alpha^i}$ for $i = 1, \ldots, N, N+2, \ldots, 2N$, where N represents the maximum user (or user group) capacity of the system and $v = g^{\gamma} \in \mathbb{G}$. The public key (provided to each user and stored in non-secure memory) is:

$$K_{\text{pub}} = (g, g_1, \ldots, g_N, g_{N+2}, \ldots, g_{2N}) \in \mathbb{G}^{2N+1}. \tag{5.1}$$

The private key for user $i \in \{1, \ldots, n\}$ is:

$$d_i = g_i^{\gamma} = v^{(\alpha^i)} \in \mathbb{G}. \tag{5.2}$$

Encryption of the group key

Encryption takes as input the set of privileged users $\mathcal{G}$ and the public key. The central station picks a random $t \in \mathbb{Z}_p$, sets the group key $K_G = e(g_{N+1}, g)^q \in \mathbb{G}_1$ and broadcasts

$$H = (C_0, C_1) = \left(g^q, \left(v \prod_{j \in \mathcal{G}} g_{N+1-j} \right)^q \right) \in \mathbb{G}^2. \tag{5.3}$$

Decryption

Decryption at user i takes as input $\mathcal{S}$, d_i, H and the public key K_{pub}. The decryption algorithm recovers K_G as follows:

$$K_G = \frac{e(g_i, C_1)}{e\left(d_i \prod_{j \in \mathcal{G}, j \neq i} g_{N+1-j+i}, C_0\right)} \tag{5.4}$$

The BMBE scheme allows to exclude any number of users from the privileged set with a message of size $\mathcal{O}(1)$. This comes at the cost of a $\mathcal{O}(N)$ public key size. However, since the public key can be stored in non-secure memory, most of application scenarios should be able to cope with this requirement. However, like most schemes for stateless receivers [12, 71], it does not scale well with the dynamics of the system: once a user leaves the system its private key will remain unused. During the system setup phase, N private keys are generated with $N > n$ to accommodate the arrival of new users. The more leave and join events, the more keys will become useless. Choosing a large value of N postpones the need for rekeying, but on the other hand it may excessively increase the public key size. The computational complexity is dominated by the $\mathcal{O}(|\mathcal{G}|)$ multiplications during the decoding phase, however if each user i pre-computes $\prod_{j\in\mathcal{G},j\neq i} g_{N+1-j+i}$, the complexity is reduced to $\mathcal{O}(m)$ divisions, with m the number of users that are revoked at each membership change.

5.3.5 Adaptation to GNSS: short and long term users

Let us suppose that some of the service users are short-term, i.e., their authorization will only last for a short amount of time with respect to the system lifetime, while others are long-term. As discussed in paragraph 5.3.2 it is assumed that at each join event the system will know the permanence time of the user. Short term users will get a private key via aiding channel, and that key will remain valid for a time slot of duration T_{min}, after which they will need to get a new key. Unplanned user eviction will not be possible for short term users, therefore T_{min} shall be chosen short enough to avoid exposing the system for long in case a short term user breaks its tamper proof device. Long term users will get private secret key material that remains valid for longer, and the system will be able to perform unplanned eviction on them, if necessary. The group keys can be organized as the intermediate values of a hash chain that starts with a seed value, S_v, or as the leaves of a binary hash tree with root S_v, following the examples from literature ([69], [16]). S_v will be encrypted through the BMBE scheme and broadcast, but only the authorized users will correctly decrypt it. Short term users will only get an intermediate value for the key, and they will therefore be able to retrieve a subset of the group keys for the interval T_{min}, according to the requested access. The architecture of the scheme is represented in Fig. 5.5.

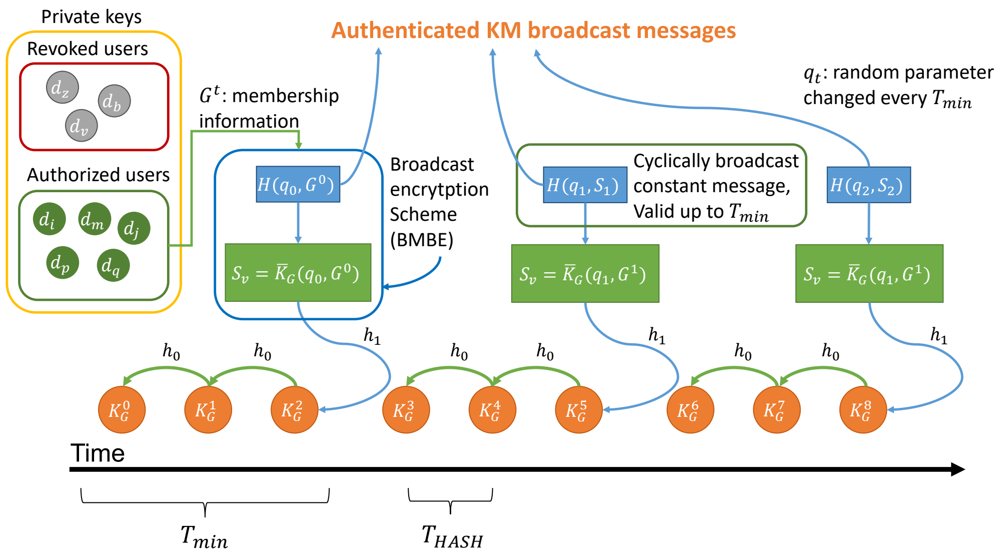

Private keys
Revoked users
d_z
d_b
d_v
Authorized users
d_i
d_m
d_j
d_p
d_q
G^t: membership information
Authenticated KM broadcast messages
$H(q_0, G^0)$
$S_v = \bar{K}_G(q_0, G^0)$
Broadcast encrytption Scheme (BMBE)
$H(q_1, S_1)$
Cyclically broadcast constant message, Valid up to T_{min}
$S_v = \bar{K}_G(q_1, G^1)$
q_t: random parameter changed every T_{min}
$H(q_2, S_2)$
$S_v = \bar{K}_G(q_1, G^1)$
h_0
h_1
K_G^0
K_G^1
K_G^2
K_G^3
K_G^4
K_G^5
K_G^6
K_G^7
K_G^8
Time
T_{min}
T_{HASH}

Figure 5.5: The architecture of the BMBE scheme.

The broadcast message H and the key K_G are represented as functions of q and $\mathcal{G}^t$, to emphasize that they depend on the authorized set at time t and on the random parameter q, that determines a key K_G^t that is different from all the previous ones. h_0 and h_1 are two different hash functions that expand the seed $S_v = \bar{K}_G$ into all the group keys used during an interval T_{min}.

5.3.6 Efficient scheduled expiration

In order to efficiently implement the function of authorization expiration, the linear structure of the scheme will be exploited. The terms $(g, g_1, g_2, \dots)$ in the public key are called *key generators*, as each of them is used to obtain one private key $d_i = g_i^{\gamma}$. They can be logically divided into time slots, consequently forcing a time order on the corresponding private keys. This subdivision is public and known by all receivers. Each user will be assigned a private key from the time slot corresponding to the expiration time of its authorization. In this way a protocol can be devised to schedule authorization expiration without having to broadcast the indexes of the expired users (and thus sparing precious bandwidth). It can be noticed that $P = \prod_{j\in\mathcal{G}} g_{N+1-j}$ is contained in the term C_1 of the broadcast message. The system revokes the authorization of user i by removing the factor g_{N+1-i} from P. In the beginning, all generators are authorized. As soon as one time slot expires, the generators linked to that time slots are removed from P at the system side. Correspondingly, receiver i will compute the product $Q_i = \prod_{j\in\mathcal{G},j\neq i} g_{N+1-j+i}$ as a part of the operations for decrypting K_G: all the expired indexes j will be removed from this computation as well. No additional broadcast information is needed in this operation.

5.3.7 Improved scalability

In order to make the scheme more scalable and usable, a protocol has been devised for re-introducing private keys owned by evicted users while preserving confidentiality. When a user's authorization expires, its private key d_i, corresponding to generator g_i in the public key, is not usable anymore, as g_{N+1-i} has been excluded from P in the message H. If g_{N+1-i} were re-introduced as a factor of P and d_i given to a new authorized user, then the former owner of d_i would be able to decode the group key as well.

The system however is parametrized by the value γ that links d_i to g: $d_i = g_i^{\gamma}$, that we can thus refer to as $d_{i,\gamma}$. Since the parameter γ influences only the term C_1 of the ciphertext and the private keys, an update of γ can be scheduled if users are provided with multiple versions of their private key $\left(d_{i,\gamma_{t_0}}, \dots, d_{i,\gamma_{t_M}}\right)$. Then an update of the parameter γ can be issued by the system by substituting $C_1(\gamma_{t-1})$ with $C_1(\gamma_t)$ and the users will correspondingly decode the group key by using d_{i,γ_t} instead of $d_{i,\gamma_{t-1}}$, which will immediately become outdated and useless. It can be noticed that d_{i,γ_t} can not be derived from $d_{i,\gamma_{t-1}}$ because of the discrete logarithm problem in finite fields.

Once the parameter γ_{t-1} is updated to γ_t, the private key $d_{i,\gamma_t} = g_i^{\gamma_t}$ can be redistributed to another user since non-authorized users that own $d_{i,\gamma_{t-1}} = g_i^{\gamma_{t-1}}$ can not retrieve the new private key nor the group key. An analogous use of γ to force a diversity in the private keys was suggested in [15] in the scope of copyright protection.

The protocol is modified as follows:

- the timeline is divided into time slots of length T_{min}. T_{min} represents the minimum time between changes in the privileged set and the maximum authorization interval for a short term user. At the beginning of each slot a new key seed $S_v = \bar{K_G}(q_t, S_t)$ will be issued.

- M adjacent time slots are grouped into *segments* of $T_{\text{med}} = MT_{\text{min}}$. T_{med} represents the time between the re-introduction of used key generators g_i in the system. Each segment is characterized by a different value of $\mathbb{G}amma_t$, and at the beginning of a new segment all generators g_i that had previously been excluded from the privileged set are re-inserted.

Fig. 5.6 represents the schedule of the authorization expiration and key reintroduction. At the end of T_1 all the corresponding key generators will be revoked, as the users who own them are past their intended permanence time. Similarly at the end of each slot, the corresponding private keys become invalid. At the end of T_4, however, all keys are reintroduced in the system, but with a different γ_t. After T_{med} (in this case $M = 4$), the same schedule repeats. It can be noticed that if a user joins at T_0 with a permanence time longer than T_{med}, the system will have to provide it with more than one key: the user will need one key for each different parameter γ_t that will be used during the permanence time, that is n_{key} keys with $n_{\text{key}} = \lceil T_P/T_{\text{med}} \rceil$. For this reason the last slot of each time segment of length T_{med} will contain more keys than the others: each user whose subscription extends for longer than a particular segment T_{med} will get one private key linked to a key generator of the last slot. This private key can be used for decoding the group key for the whole duration of the segment (T_{med}). Thus the last slot of a segment is linked to keys for long term users that will be authorized for the whole duration of the slot, while users whose subscription ends before the end of the segment will get a key from one of the internal slots. In case the average permanence time is QT_{med} and assuming that join events are uniformly distributed in time, the last slot of any segment needs to contain Q times the number of keys of any other slot.

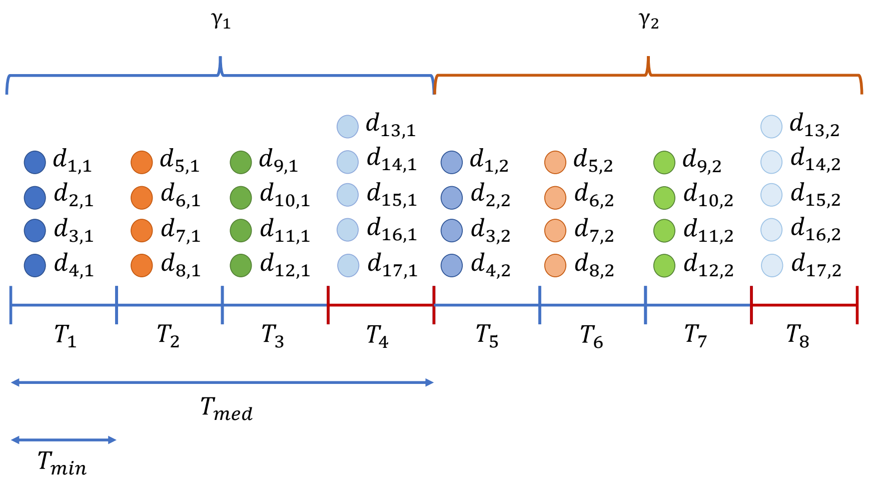
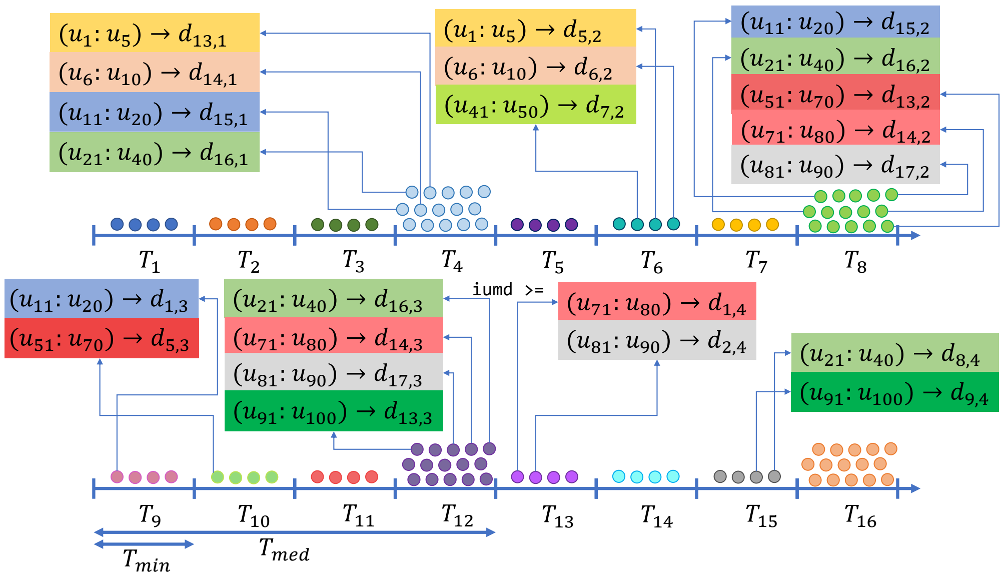

γ1
γ2
$d_{1,1}$ $d_{2,1}$ $d_{3,1}$ $d_{4,1}$
$d_{5,1}$ $d_{6,1}$ $d_{7,1}$ $d_{8,1}$
$d_{9,1}$ $d_{10,1}$ $d_{11,1}$ $d_{12,1}$
$d_{13,1}$ $d_{14,1}$ $d_{15,1}$ $d_{16,1}$ $d_{17,1}$
$d_{1,2}$ $d_{2,2}$ $d_{3,2}$ $d_{4,2}$
$d_{5,2}$ $d_{6,2}$ $d_{7,2}$ $d_{8,2}$
$d_{9,2}$ $d_{10,2}$ $d_{11,2}$ $d_{12,2}$
$d_{13,2}$ $d_{14,2}$ $d_{15,2}$ $d_{16,2}$ $d_{17,2}$
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8
T_{med}
T_{min}

Figure 5.6: An example of the time organization of the keys.

In order to better explain the key distribution phase, an illustrative example is represented in Fig. 5.7, where one key is assumed to be associate with a group of users instead of a single user.

The proposed key distribution and key validity schedule allows to recover expired slots in the linear structure of the scheme, and thus free slots for new users. This is a practical solution to the scalability issues that affect most key management schemes,



$(u_1: u_5) \to d_{13,1}$
$(u_6: u_{10}) \to d_{14,1}$
$(u_{11}: u_{20}) \to d_{15,1}$
$(u_{21}: u_{40}) \to d_{16,1}$
$(u_1: u_5) \to d_{5,2}$
$(u_6: u_{10}) \to d_{6,2}$
$(u_{41}: u_{50}) \to d_{7,2}$
$(u_{11}: u_{20}) \to d_{15,2}$
$(u_{21}: u_{40}) \to d_{16,2}$
$(u_{51}: u_{70}) \to d_{13,2}$
$(u_{71}: u_{80}) \to d_{14,2}$
$(u_{81}: u_{90}) \to d_{17,2}$
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8
$(u_{11}: u_{20}) \to d_{1,3}$
$(u_{51}: u_{70}) \to d_{5,3}$
$(u_{21}: u_{40}) \to d_{16,3}$
$(u_{71}: u_{80}) \to d_{14,3}$
$(u_{81}: u_{90}) \to d_{17,3}$
$(u_{91}: u_{100}) \to d_{13,3}$
iumd >=
$(u_{71}: u_{80}) \to d_{1,4}$
$(u_{81}: u_{90}) \to d_{2,4}$
$(u_{21}: u_{40}) \to d_{8,4}$
$(u_{91}: u_{100}) \to d_{9,4}$
T_9 T_{10} T_{11} T_{12} T_{13} T_{14} T_{15} T_{16}
T_{min}
T_{med}

Figure 5.7: An example of key distribution per user groups. The following join events are considered:

- 40 users join the system at the beginning of the first T_{med}, i.e., at the beginning of T_1:
  - u_1 to u_{10} with permanence time $T_{\text{med}} + 2T_{\text{min}}$;
  - u_{11} to u_{20} with permanence time $2T_{\text{med}} + T_{\text{min}}$;
  - u_{21} to u_{40} with permanence time $3T_{\text{med}} + 3T_{\text{min}}$;
- 50 users join the system at the beginning of the second T_{med}, i.e., at the beginning of T_5:
  - u_{41} to u_{50} with permanence time $2T_{\text{min}}$;
  - u_{51} to u_{70} with permanence time $T_{\text{med}} + 2T_{\text{min}}$;
  - u_{71} to u_{90} with permanence time $2T_{\text{med}} + T_{\text{min}}$;
- 10 users join the system at the beginning of the third T_{med}, i.e., at the beginning of T_9:
  - u_{91} to u_{100} with permanence time $T_{\text{med}} + 3T_{\text{min}}$.

and it comes at the cost of additional secret storage requirements for the private keys, depending on the permanence time of each user and on the choices of the system for the values of T_{med} and T_{min}.

5.3.8 Unplanned user eviction

In case of unplanned eviction an additional message is needed, containing information on which generators have been removed from the privileged set before schedule. The size of the identification set message has to be bounded: one solution is that of *limiting the statelessness of the receivers* by fixing the maximum time T_{max} between two successive connections of each user to the signal in space. This allows the system to ensure that if all users respect the policy, the identification set information that has been broadcast since $t - T_{\text{max}}$ has been received. The system can thus suspend the broadcast of old information, compressing the additional overhead. Moreover, in case this mechanism is not enough to contain the size of the identification set message, the policy can be enforced by limiting the size of the additional overhead to a maximum length of U_{max} bits. The system will then evaluate whether it is more convenient to perform new user eviction operations right away (e.g., in case of suspected key compromise) or to wait for the identification set message to shrink (e.g., when users are not respecting the subscription terms but the system is not in danger). In case the additional overhead reaches the size U_{max} and new user eviction operations are needed right away, the system will broadcast an alert message, notifying the users that they should connect to an aiding channel to download the identification set information. The system shall be designed in such a way that the probability of this event is extremely low: the tamper resistant devices need a high enough level of security.

Two operation modes can be defined for the system, namely the *regular mode*, in which all key generators are removed and re-introduced according to the normal schedule, and the (*unplanned eviction mode*), when an identification set message is broadcast together with the regular message of the BMBE scheme. Apart from the identification set message (that causes a longer broadcast message), the unplanned eviction mode is identical to the regular mode in the key schedule, except for the generators that have been revoked before the natural expiration: those will never be reintroduced in the system. Re-introduction of generators revoked before schedule takes longer than T_{med} and therefore it may not be worth implementing; however a thorough definition of this function is left for future work.

5.3.9 Continuity of service

The devised protocol allows the system to schedule the change of group keys, issuing a new one every T_{min}. The actual time in which the system starts broadcasting the new KMM (i.e., the whole key management information needed to retrieve the group key) in place of the old one should be scheduled in order to preserve the *continuity of service* for receivers that are using the service at that time. The demodulation of the KMM may take several minutes, therefore the dissemination of key management data related to the new group key shall be broadcast some time before the key switching instant, in order to provide continuity to all connected receivers. Service providers should estimate a reasonable upper bound for the decoding time, $\mathrm{T}_{\mathrm{dec}}^{\mathrm{max}}$, accounting for channel errors. Key management messages for the new group key will be broadcast $\mathrm{T}_{\mathrm{dec}}^{\mathrm{max}}$ before the scheduled group key update. This policy will guarantee continuity with the desired probability.

5.3.10 Theoretical securtiy analysis

The security of the broadcast encryption system presented in [15] is related to the hardness of the so-called *decision ℓ-BDHE problem*. Given two groups $\mathbb{G}, \mathbb{G}_1$ of prime order p, the bilinear map $e : \mathbb{G} \to \mathbb{G}_1$, the integer ℓ and the vector

$$\left(h, g, g^{\alpha}, g^{\alpha^2}, \dots, g^{\alpha^{\ell}}, g^{\alpha^{\ell+2}}, \dots, g^{\alpha^{2\ell}}\right) \in \mathbb{G}^{2\ell+1}$$

where $g, h \in \mathbb{G}$ are generators and $\alpha \in \mathbb{Z}_p$ is unknown, the problem requires to decide whether $e(g,h)^{\alpha^{\ell+1}} = k$ holds or not, given $k \in \mathbb{G}_1$. Note that this requirement is stricter than the hardness of the corresponding computational problem, which asks to compute $e(g,h)^{\alpha^{\ell+1}}$, given the same inputs. The decision (t, ε, ℓ)-BDHE assumption holds in $(\mathbb{G}, \mathbb{G}_1, e)$ if no algorithm can succeed in solving the decision ℓ-BDHE problem in less than time t with a probability larger than $1/2 + \varepsilon$. In particular, given the maximum number of authorised users N, [79, Theorem 3.1] states that the BMBE scheme is (t, ε, N) semantically secure, if the decision (t, ε, N)-BDHE assumption holds in $(\mathbb{G}, \mathbb{G}_1, e)$. Therefore, it is possible to get some information on the security parameters by studying the performance of adversaries that attempt to solve the decision N-BDHE problem. This approach leads to the bounds 5.6 and 5.7.

Generic group model

The *generic group model*, first defined in [80], deals with adversaries that do not exploit any special feature of the particular group in their attacks. Assume that attack algorithms behave as Turing machines, observing bit strings instead of group elements. They can store information, but they cannot compute group operations or pairings on their own: these operations are provided by oracles.

Definition 5.3.1. Consider the groups with pairing $(\mathbb{G}, \mathbb{G}_1, e)$ and let $m = O(\log_2 |\mathbb{G}|) \in \mathbb{N}$ be the length of binary encodings. A *generic pairing-based algorithm* $\mathcal{A}$ is a probabilistic algorithm that behaves as follows.

- It takes as input two encoding lists, containing binary encodings of elements in $\mathbb{G}$ and $\mathbb{G}_1$:

$$L_{\mathbb{G}} = (\sigma(g_1), \dots, \sigma(g_s))$$
$$L_{\mathbb{G}_1} = (\rho(h_1), \dots, \rho(h_t))$$

  where $g_i \in \mathbb{G}$, $h_j \in \mathbb{G}_1$ and $\sigma(g_i), \rho(h_j) \in \{0,1\}^m$ are random binary strings, for $i = 1, \dots, s$ and $j = 1, \dots, t$.

- During the execution, it has access to three oracles.

- An oracle for the group operation in $\mathbb{G}$: it takes as input two strings $\sigma(g_i), \sigma(g_j)$, encodings for $g_i, g_j \in \mathbb{G}$, from the updated encoding list relative to $\mathbb{G}$. Then the oracle outputs a binary string $\sigma(g_i \cdot g_j)$ or $\sigma(g_i \cdot g_j^{-1})$ in $\{0,1\}^m$, according to the query, which is then appended to $L_{\mathbb{G}}$
- An oracle for the group operation in $\mathbb{G}_1$: it takes as input two strings $\rho(h_i), \rho(h_j)$, encodings for $h_i, h_j \in \mathbb{G}_1$, from the updated encoding list relative to $\mathbb{G}_1$. Then the oracle outputs a binary string $\rho(h_i \cdot h_j)$ or $\rho(h_i \cdot h_j^{-1})$ in $\{0,1\}^m$, according to the query, which is then appended to $L_{\mathbb{G}_1}$.
- An oracle for the bilinear mapping $e : \mathbb{G} \to \mathbb{G}_1$: it takes as input two strings $\sigma(g_i), \sigma(g_j)$, encodings for $g_i, g_j \in \mathbb{G}$, from the updated encoding list relative to $\mathbb{G}$. Then the oracle outputs a binary string $\rho(e(g_i, g_j)) \in \{0,1\}^m$, which is then appended to $L_{\mathbb{G}_1}$.

Moreover, both $\mathcal{A}$ and the oracles keep track of outputs during the simulation.

- It outputs a binary string, denoted by $\mathcal{A}(L_{\mathbb{G},s}, L_{\mathbb{G}_1,t}) \in \{0,1\}^*$.

The main limitation of the generic approach is that some specific algorithm may exist that exploits features of some particular group or bilinear pairing to achieve some more efficient attack. However the analysis carried out in such abstract framework allows us to derive some information on the hardness of the considered problem. Such techniques are currently used to study the security of several encryption systems in the literature [79, 81–83]: they provide evidence that an encryption system's security can be safely based on a certain cryptographic problem as long as all known algorithms for solving it are generic. As soon as some specific algorithm is devised, then new security parameters need to be chosen in such a way that the new algorithm has little or no advantage with respect to the generic ones. Further comments on this model and its validity can be found in [84].

Hardness of the decision ℓ-BDHE problem in the generic group model

Let us assume there exists a generic pairing-based algorithm $\mathcal{A}$ attacking the ℓ-BDHE problem; we can derive an upper bound to its probabilistic advantage 5.6. In order to study the cryptographic problem it is necessary to state it in a more general version using polynomials, that once evaluated give a particular instance of the above problem. In this case, the adversary takes as input lists of binary encodings, as in Definition 5.3.1, corresponding to the polynomials in $\mathbb{Z}_p[X, Y]$ contained in the following vectors:

$$\begin{aligned} &U(X,Y) = (1, Y, X, X^2, \ldots, X^{\ell}, X^{\ell+2}, \ldots, X^{2\ell}), \\ &V(X,Y) = (1). \end{aligned} \tag{5.5}$$

Let $f(X,Y)$ be the polynomial $X^{\ell+1}Y \in \mathbb{Z}_p[X,Y]$. Polynomials in U, V, once evaluated at some $x, y \in \mathbb{Z}_p$, can be used as exponents to generate elements in $\mathbb{G}, \mathbb{G}_1$, given generators $g \in \mathbb{G}$ and $h := e(g,g) \in \mathbb{G}_1$ of both groups. Therefore, each binary encoding is associated to an element in those groups, i.e. to h and to the elements obtained as $g^{u(x,y)}$, where $u(X,Y)$ is an entry of $U(X,Y)$. Assume that $\mathbb{G}, \mathbb{G}_1$ are both finite groups of prime order

p, as before. Then, given such initial encodings as input, the adversary has to distinguish among the two encodings of $h^{f(x,y)}$ and h^t, for a given $t \in \mathbb{Z}_p$, the one corresponding to the former element. As explained in [85], it is possible to apply [79, Theorem A.2] to the case of the polynomial decision ℓ-BDHE problem, getting:

$$\left| \Pr\left[\mathcal{A}\begin{pmatrix} \sigma(g^{U(x)}), \rho(h), \\ \rho(h^{t_0}), \rho(h^{t_1}) \end{pmatrix} = b \right] - \frac{1}{2} \right| \leq \frac{2\ell(q+2\ell+4)^2}{p}, \tag{5.6}$$

where q is the maximum number of queries allowed to the generic algorithm $\mathcal{A}$ and where $x \in \mathbb{Z}_p^2$, $t \in \mathbb{Z}_p$, $b \in \{0,1\}$ are chosen uniformly at random. Moreover, we define $t_b := f(x,y)$, $t_{1-b} := t$. The left hand side of the above inequality is the probabilistic advantage of $\mathcal{A}$. This result gives a lower bound on the complexity of such adversaries, where the time is measured by the number of queries. Indeed, given an advantage $\varepsilon \in [0, 1/2]$, then the inequality 5.6 gives, after few computations:

$$q\mathbb{G}e\sqrt{\frac{p\varepsilon}{2\ell}} - 2\ell - 4 =: q_{\text{min}}(p,\varepsilon,\ell). \tag{5.7}$$

Thus $q_{\text{min}}(p,\varepsilon,\ell)$ is a lower bound on the number of queries needed to achieve advantage at most ε and we conclude that any generic adversary, achieving such advantage ε, must take time at least $\Omega(q_{\text{min}}(p,\varepsilon,\ell))$. In particular, the security of the cryptosystem described in section Sec. 5.3.4 is based on the decision N-BDHE problem. Thus, applying the above analysis to [79, Theorem 3.1] it follows that the semantic security with parameters (t,ε,N) is achieved for every $t < q_{\text{min}}(p,\varepsilon,N)$, where the time is measured as the number of issued queries.

Elliptic curves for implementation

The hardness of the ℓ-BDHE problem in $(\mathbb{G}, \mathbb{G}_1, e)$ is not the only concern when investigating the security of the proposed system. It should be noticed that the bilinear map $e : \mathbb{G} \to \mathbb{G}_1$ is usually defined on groups of points over elliptic curves. Therefore, the choice of such curves is especially important, since it affects the security of the entire system. A suitable elliptic curve E over a field $\mathbb{F}_q$ should admit pairings giving values in sufficiently large finite fields, such that the MOV reduction [86] is ineffective. The MOV attack to the discrete logarithm problem (DLP) over the elliptic curves works by solving an instance of the same problem on the finite field containing the image of a bilinear pairing. Indeed, if $(\mathbb{G}, \mathbb{G}_1, e)$ are groups with a pairing as above, then given $g_1, g_2 \in \mathbb{G}$ of order p there exists α such that $g_2 = g_1^\alpha$. Therefore, $h_1 := e(g_1, g_1) \in \mathbb{G}_1$ and $h_2 := e(g_2, g_1) = h_1^\alpha \in \mathbb{G}_1$ have order p and it is possible to find α by solving the DLP in $\mathbb{G}_1$, which is contained in some finite field. Notice that in the finite field case the DLP can be solved much more efficiently than in the elliptic curve case. Recall that the group of $\mathbb{F}_q$-rational points of E, denoted by $E(\mathbb{F}_q)$, contains all points with coordinates in $\mathbb{F}_q$ laying on the curve. In particular, the following conditions should hold:

- the DLP must be computationally infeasible in the cyclic subgroup $E(\mathbb{F}_q)[p] \leq E[p]$, which contains all p-torsion points with coordinates in $\mathbb{F}_q$, for some prime number $p \mid m = |E(\mathbb{F}_q)|$ such that $(p, \text{char}(\mathbb{F}_q)) = 1$;

- the DLP must be computationally infeasible in $\mathbb{F}^*_{q^k}$, where k is the embedding degree of E with respect to p.

The former requirement is achieved when p is a large prime factor of m, while the latter depends on the embedding degree. Definitions and properties of the elliptic curves can be found in [87]. Moreover, since the cryptosystem of our interest is based on symmetric bilinear maps, only supersingular elliptic curves allow its implementation. As these curves admit embedding degrees $k \in \{1, 2, 3, 4, 6\}$, there is an additional constraint on the parameter k. According to literature [88–90], it is better to avoid elliptic curves defined over finite fields of characteristic 2 and 3. The remaining supersingular curves have only embedding degree $k = 2$ or $k = 3$. In [91] a family of supersingular elliptic curves that allow secure implementation is proposed, under the choice of suitable parameters. The advantage of such curves is the efficient calculation of the bilinear map, provided that the reduced Ate maps [92] are used. Given a prime $r > 3$, such that $r \equiv 5 \mod 6$, let $b \in \mathbb{F}_{r^2}$ be a square, but not a cube. Then define the elliptic curves E_b by means of the affine equation

$$y^2 = x^3 + b.$$

The group of $\mathbb{F}_{r^2}$-rational points on all curves E_b has cardinality

$$|E_b(\mathbb{F}_{r^2})| = r^2 - r + 1.$$

For the sake of security and efficient pairing computation, consider the largest prime divisor p of $|E_b(\mathbb{F}_{r^2})|$ such that $p^2 \nmid |E_b(\mathbb{F}_{r^2})|$; then the pairing should be defined on the group $G = E_b(\mathbb{F}_{r^2})[p]$. The embedding degree of E_b with respect to p is $k = 3$. Therefore a prime r should be chosen, such that the DLP is computationally infeasible in $\mathbb{F}_{r^6}$. According to [91], the prime characteristic r should be at least 200 bits long, in order to avoid the Gaudry–Hess–Smart attack (GHS) [93]. Moreover, according to [89] and [83], the characteristic r of the underlying finite field should be chosen while taking into account the following asymptotic computational complexity for the DLP solution in $\mathbb{F}_{r^6}$:

$$\exp \sqrt[3]{\left(\frac{384}{9} + o(1)\right) \log r (\log(6 \log r))^2}, \tag{5.8}$$

where $o(1)$ is for $r \to \infty$.

In conclusion, the security analysis of the BMBE of our interest can be split into two parts. On one side we have achieved a bound on the asymptotic complexity of a generic pairing-based attack algorithm having at most advantage ε. Notice that the number of queries made by the adversary are a polynomial function of the bit length m of the encodings, since it is reasonable to allow only polynomial time to the attacker. Moreover, the prime p must be an exponential function of m, for security reasons. Eventually the integer N is a design parameter, which is decided taking into account the number of users that subscribe the service, and thus is fixed. Therefore one can choose m and ε such that the system has high semantic security. On the other hand, when choosing the elliptic curves for the implementation of the system one has to choose a suitable prime r, that gives the dimension of the finite field on which a curve is defined. The value of r^2, should be at least 200-bit long and, in addition, r should be such that the computational complexity 5.8 makes the MOV attack infeasible, as described above. Recall that the choice of r is related to the cardinality p of $\mathbb{G}$, the latter being the greatest prime divisor of $r^2 - r + 1$. As an example [91] propose an elliptic curve of the above family, where $\log r = 522$ and the cardinality of the finite field $\mathbb{F}_{r^6}$, containing the image of the bilinear pairing, is 3132 bits long. Notice that elements in $\mathbb{G}$ are $\mathbb{F}_{r^2}$-rational points, therefore their coordinates are contained in $\mathbb{F}_{r^2}$; the bilinear pairing outputs roots of unity in $\mathbb{F}_{r^6}$, which are elements of $\mathbb{G}_1$. By considering the computational complexity 5.8, the choice of this elliptic curve's parameters guarantees the 128-bits security with respect to the MOV attack. Another option, however, is that of dropping the value of $\log r$ up to 256 bits, accepting looser security requirements; this allows to shorten broadcast communications.

5.3.11 Application to Galileo commercial Service

Galileo's E6 signal is composed by a data (E6-B) and a pilot component (E6-C). Both spreading codes have higher chipping rate (5.115 Mchip/s) than the Open Service and support spreading code encryption. To the best of the authors' knowledge no official documentation exists on the services that will be provided through E6-B, yet. For this reason the access control scheme that regulates the provision of services shall be flexible enough to accommodate whatever new concept may become interesting in the future. As an example, the literature suggest Precise Point Positioning (PPP) as a promising candidate, offering centimeter level precision through the dissemination of clock and orbit correction at a higher rate, or signal authentication through spreading code encryption [94]. However, it is worth underlining that the aim of the key management scheme devised in this paper is to protect the access to data, regardless of the details of the service provision. The underlying service can thus be seen as a black box that requires a fresh new key, N_b bits long, every T_b. These bits must be encrypted with the group key in order to regulate the access to the service. In this work we adapt the access control scheme to Galileo's E6-B, assuming that no spreading code encryption is employed on this component. However, it is worth remarking that there is no reason not to apply the proposed scheme to a different component, or even to a different scenario with the same kind of constraints. The KMM may even be broadcast on a different component with respect to the one that carries the protected service.

Fig. 5.8 shows the page structure of E6-B [94]. The 448 bits of data broadcast will be mostly available for the service broadcast, leaving only a small fraction to KMMs.

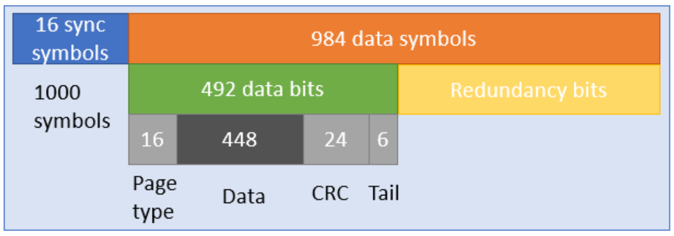


Figure 5.8: Structure of the E6-B page, [94].

The key management message

In the regular operating mode the overhead of the system consists of the message:

$$H = (C_0, C_1) = \left(g^q, \left(v \prod_{j \in \mathcal{G}} g_{n+1-j} \right)^q \right) \in \mathbb{G}^2. \tag{5.9}$$

that are two elements of a bilinear group of prime order p.

According to [95], acceptable lower bounds for bilinear maps implemented with *Weil pairings* are 160 bits for elements of $\mathbb{G}$, as C_1 and C_2, and 1024 bits for the elements of $\mathbb{G}_1$ as K_{G}. However, since 128 bits are recommended an adequate security level for the near future, as stated by NIST [96] and ENISA [66], it is more appropriate to consider 256 bits for $\mathbb{G}$ and 3072 bits for $\mathbb{G}_1$. The group key K_{G} which can be obtained by authorized users is thus 3072-bit long, but it can be hashed by SHA-256 to obtain a 256-bit key while preserving the 128 bits of security of the scheme. The other group keys of the chain will be obtained through the same hash function, SHA-256, applied multiple times: the last key obtained through the one-way chain will be the first to be used as data-encryption symmetric key. Each of these keys will provide 128-bits of security. As stated in [11], the length of the hash chain, which is used to provide symmetric keys with shorter cryptoperiod, must be limited to control the collision probability along the one-way chain. There is a trade-off between the cryptoperiod (i.e., the time of validity and usage) of the group key and the length of the one-way chain: a shorter cryptoperiod improves security, but long chains can lead to a loss with respect to the original security level.

In order to avoid attackers tampering on the key management data, a layer of authentication is needed. In this paper we opt for the well-experimented elliptic curve cryptography, specifically 256-ECDSA with a signature of 512 bit length (two elements of a group, each 256 bit long). The chosen signature scheme is standardized and used by several security protocols, being a good compromise between security and signature length.

With these considerations, the length of the KMMs in the regular mode is 1024 bit (two 256 bit elements for access control and two for authentication), while in unplanned eviction mode the identification set message must be added to the count. The additional overhead will be limited to $U_{\max} = 1000$ bits as an example application, leading to a total communication overhead of 2024 bits. Fig. 5.9 shows the revocation capabilities guaranteed by this choice in the worst case, with respect to the number of key generators, N (the maximum user or user groups capacity).

The KMM can and shall be sent in the clear: encryption would only create a further key distribution problem.

5.3.12 Performance Evaluation

In the following the performance of the scheme will be evaluated by considering an arbitrary but realistic example application. The proposed scenario is based on reasonable assumptions and educated guesses coming mainly from the content of [97] and [94] and its purpose is solely illustrative. In [97] it is reported that the European Commission

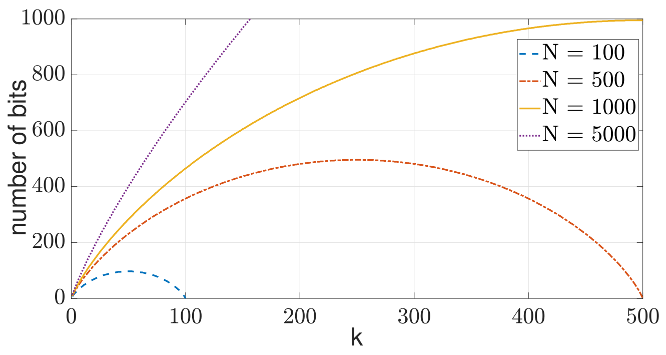


Figure 5.9: number of bits required for the identification set message to identify k out of N_g users or user groups.

has decided to provide the High Accuracy Service (HAS) openly and free of charge, with a precision of 20cm. On the other hand, a Commercial Authentication Service will be dedicated to paying users. Under these considerations it is assumed that most of the commercial service data rate will be dedicated to data dissemination, while key management will occupy a smaller number of bits per page. In [94] it is stated that a high accuracy service would likely require around 75% of the bandwidth in order to cope with the accuracy requirements, while authentication data may be accommodated in what is left of the 448 bits in each page of the commercial service data component. In order to reasonably fit the key management data, let us suppose to allocate 28 bits per page for access control and the remaining 420 bits for service data. We assume to reserve 380 bits for high accuracy and 40 bits for authentication, however it is important to remark that the choice of the type of service and the allocation does not influence the access control scheme and should be left to the system designer.

The two KM modes have different lengths for the KMM: in the regular mode the total overhead is 1024 bits, while in the unplanned eviction mode it is 2024 bits. The KM message of each provider can be transmitted in 37 pages in the best case (regular mode), and 73 pages in the worst case (unplanned eviction mode). Each page has its own CRC and FEC and faces channel errors, leading to a page error rate P_e. In order to maximize the probability of error-free demodulation, each key management message should be concentrated on the least possible number of pages. The discussed example is represented in Fig. 5.10.

Before being allowed to use the service, each receiver needs to receive 37 to 73 pages. The decoding performance of the receivers are significantly affected by fading, that in turn depends on the elevation of the satellites, the type of environment and the speed of receivers.

In [98] these effects are simulated through the use of the 2-state LMS model, which is a generative model that outputs time series according to a combination of distributions. The propagation is simulated according to the intensity of shadowing events, which lead to the system being either in a "good state" or a "bad state", each characterized by specific parameters.

As in this works we aim at providing an idea of the decoding performances, the results of the field tests reported in [98] will be solely exploited to derive some meaningful

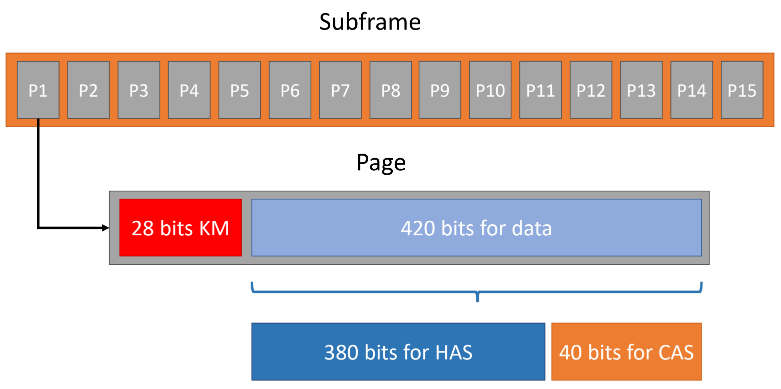


Figure 5.10: Proposed allocation of the KMM bits.

numbers for the page error rate (PER). The test campaign reports that the PER is highly sensible to the satellite elevation and the propagation conditions in urban and suburban environments. Even in the rural scenario, the PER is around 10^{-1} for low elevations, improving to 10^{-2} or 10^{-3} for medium to high elevations. The most meaningful values for the PER, that correspond to realistic C/N_0 values in most scenarios, span roughly from 0.9 to 10^{-3}. This range of values will be considered for the scope of performance evaluation.

5.3.13 Evaluation of the decoding performance

In the following we estimate the time required to decode a key management message and access the corresponding service. Let us assume the transmission of each page is subject to errors independently of the others, with a probability P, which can be derived from the experimental results in [98]. Let's assume that a receivers switches on at the beginning of the transmission (without any loss of generality) and needs to demodulate the KMM. Each page is correctly decoded with probability $(1 - P)$ independently of the others. Each satellite broadcasts all the KM information cyclically and after the first transmission of the KMM is completed, the receiver may have missed some pages. At the next re-transmission it will ignore the pages it has already decoded and try to retrieve the missing pages, and so on, until all pages have been correctly received. This can be seen as a sequence of repeated trials: the measure of interest is the time it takes for the receiver to correctly decode all N_{page} pages.

Let $A_{i,n}$ denote the event that at transmission i, page n is received in good conditions so that it can be correctly decoded. The events $A_{i,n}$, with $n = 1, \ldots, N_{\text{page}}$ and $i = 1, \ldots,$ are all independent with the same probability $P[A_{i,n}] = 1 - P$. Then, the probability that the receiver can collect all pages within the first k transmissions, that is the CDF of the random variable T (number of transmissions necessary to decode the KM message), is given by:

$$P[T \leq k] = P\left[\bigcap_{n=1}^{N_{\text{page}}}\left(\bigcup_{i=1}^{k} A_{i,n}\right)\right] \tag{5.10}$$

$$= \prod_{n=1}^{N_{\text{page}}}\left(1 - \prod_{i=1}^{k}\left(1 - P\left[A_{i,n}\right]\right)\right) \tag{5.11}$$

$$= \left(1 - P^{k}\right)^{N_{\text{page}}} \tag{5.12}$$

Correspondingly, we can express the number of trials $T_{\hat{P}}$ needed to correctly decode the KMM a high enough probability (e.g., $\hat{P} = 99\%$, corresponding to T_{99}). Once the number of transmissions is known, an approximation of the time to correctly decode it can be computed as

$$t_D \simeq N_{\text{page}} \cdot T_{99} \tag{5.13}$$

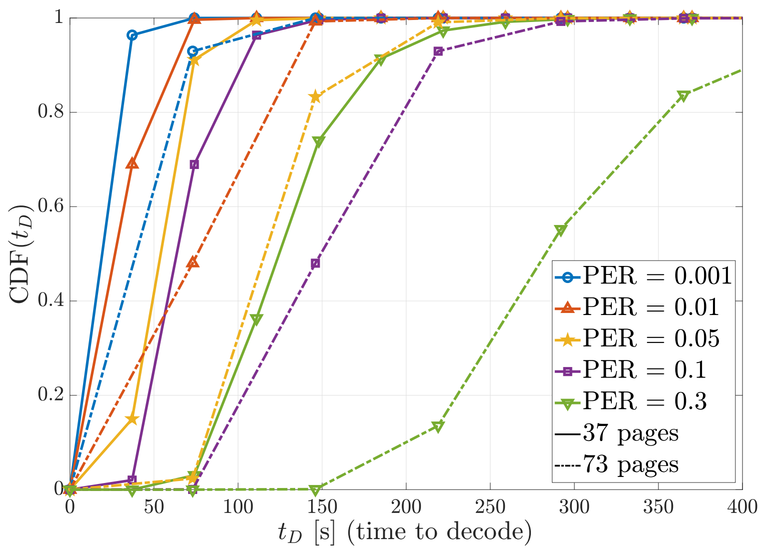


Figure 5.11: CDF for the time to decode in the best and worst case.

Fig. 5.11 reports the CDF of the time to decode for different values of the page error rate.

- If PER < 0.01, in the regular mode decoding will require less than 1.5 minutes, while the unplanned eviction mode will take 2.5 minutes.
- If $10^{-2} <$ PER $< 10^{-1}$, in the regular mode 2 minutes are likely to be enough to decode, while in the unplanned eviction mode decoding may take up to 5 minutes.
- If PER $\simeq 0.3$, in the regular mode t_D is still lower than 5 minutes, while in the unplanned eviction mode it can grow larger than 7 minutes.

Notice that the *worst case* can be tuned by service providers in order to cope with the dynamics of the channel and achieve the desired trade off between time to decode and unplanned eviction capabilities.

5.4 Conclusions

This chapter has tackled the design of secure and efficient key management architectures for two higher layer services: authentication and access control. Several solutions from literature have been assessed, while taking into account the different requirements of the GNSS context and the different security features linked to the specific use case. The proposed schemes exploit validated and well known mechanisms that are already in use in other communications systems, combined together in a secure architecture. These mechanisms are:

- multilevel PKIs, that enhance the system robustness against key compromise;
- one-way functions or chains, that allow to renew the secret key more often while maintaining a low communication overhead;
- symmetric key encryption with delayed-key-release, allowing shorter keys for the same security level;
- bilinear pairings, whose properties allow to build communication-efficient and secure cryptoschemes.

In order to tailor the key management schemes to the GNSS applications, the unique features of each scenario were exploited in order to maximize efficiency for the same security level. As an example, the low data rate requirement was achieved by offloading part of the information to the device storage in Sec. 5.1. In Sec. 5.3 the constant communication overhead in nominal mode was achieved by exploiting the knowledge on the subscriptions' expiration time. An analytic framework for evaluating the performance of the schemes in terms of decoding time was presented and validated by simulation results. The devised architectures are flexible and modular and can thus be adapted to different GNSS constellations, or even to other broadcast channels.

Chapter 6

Key Distribution Protocol over a GNSS Constellation Leveraging Intersatellite QKD Links

Free-space optical communications are an attractive alternative to RF communication links as they are characterized by higher antenna gain, higher data rates, EMC/EMI immunity, less frequency regulation issues, etc [99]. This technique is a valuable tool for defense applications, as it has intrinsic robustness against interception, eavesdropping and jamming, thanks to the narrow beamwidth [100]. It is reported in [99] that a number of space missions have successfully tested optical communication techniques in space with excellent performances. This promising results have encouraged additional developments, such as the exploitation of quantum communications in free space optical links for the purpose of authentication and integrity protection, which are particularly relevant in GNSS applications.

Narrowbeam optical communications may not provide a sufficient level of security, as the reduced beamwidth does not completely eliminate the risk of eavesdropping: when such long distances are involved (thousands of kilometers), the beam tends to spread over a larger area.

In free-space optical links the information unit is the bit, which is carried by pulses containing a big number of photons. On the other hand, quantum links refer to free-space optical communication links that are described with quantum optics and quantum information theory. The information unit is the quantum bit, or *qubit*, which is a bit of information stamped in a quantum physical property, such as the polarization of a photon. Contrarily from classic optical links, in quantum links the information is carried by weak pulses, containing few photons. Quantum links exploit properties such as the superposition of states and entanglement, which has lead to innovative methods (e.g., quantum key distribution) that are more powerful than their classic counterparts.

As the impact of an attacker hijacking and controlling a GNSS satellite may be catastrophic, adopting quantum links for satellite-to-ground and satellite-to-satellite communications allows to generate unconditionally secure keys and establish an unconditionally secure satellite network. This cryptographic feature is stronger that computational security, which depends on the computational capabilities of the attacker. Unconditional security allows to obtain the desired level of security *regardless of the attacker's computational capabilities*.

In [99] a geometric feasibility analysis is carried out for the implementation of optical quantum links among Galileo satellites. The Galileo constellation comprises 24 satellites orbiting in three orbital planes with 56^o relative inclination angle. Given the strong dependence of the achievable key rates from the inter-satellite distance, it is reasonable to assume that only pairs of satellites that get close enough sometime during their orbital period will be able to establish a sufficient amount of shared secret key bits. The simulation results on the relative distance between SVs indicate that a window of approximately 48 minutes exists in which each satellite is closer than 10000km to two SVs for each of the other orbital planes. Thus each SV has four partially overlapping time windows in which a quantum link can be exploited to exchange unconditionally secret bits.

Quantum links thus allow to build a graph representing inter-satellite connections. While it is desirable to have a high degree of connectivity in the satellite network, the constraints on the mass and volume of SVs may not allow to equip satellites with multiple transceivers. A solution is proposed in [99] that only requires one quantum transceiver, one dynamic telescope and one quantum transmitter and receiver per SV. In [99] a possible scheduling of quantum transmissions was proposed (represented in Fig. 6.1), based on a model with 9 satellites per orbital plane (as predicted in 2013 for Galileo). The model considered in this analysis, however, is based on 24 satellites.

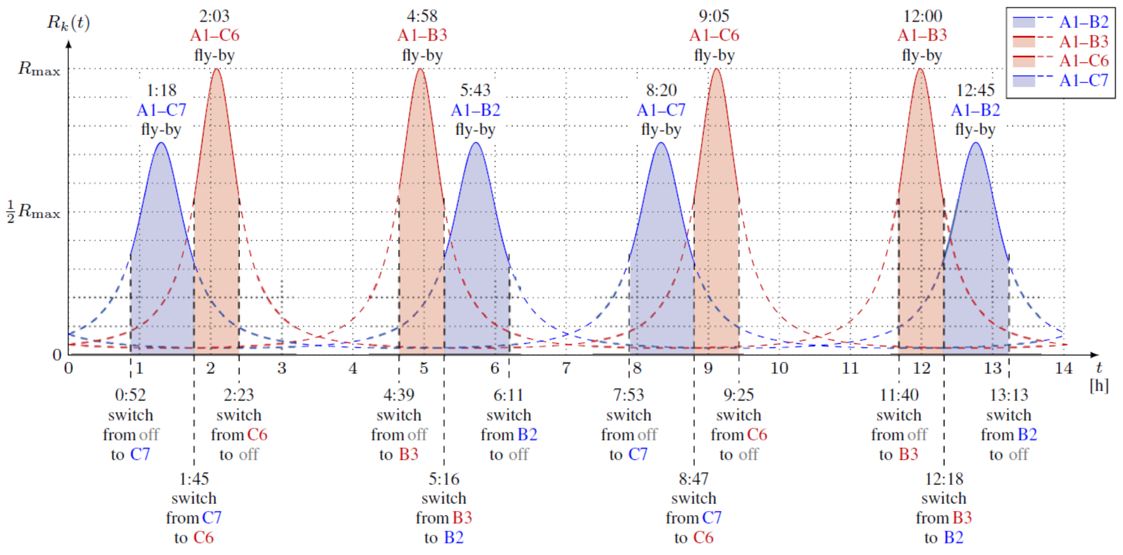


Figure 6.1: Link switching sequence proposed in [99] for satellite A1. The shaded areas represent the total secret key length achievable in one period. The original model assumed 9 satellites per orbital plane, differently from the 8 considered in this analysis.

The network graph is represented in Fig. 6.2, where the two different link types are highlighted in different colors.

The resulting graph is a regular graph where all nodes have degree four.

In order to make the graph connected an unconditionally secure key agreement protocol can be devised that exploits the existing secure connections to share key pairs among non-adjacent satellites. To devise such a protocol, the achievable SKR per orbital period for *long* and *short* links must be derived. In [99] an upper bound on the SKR is

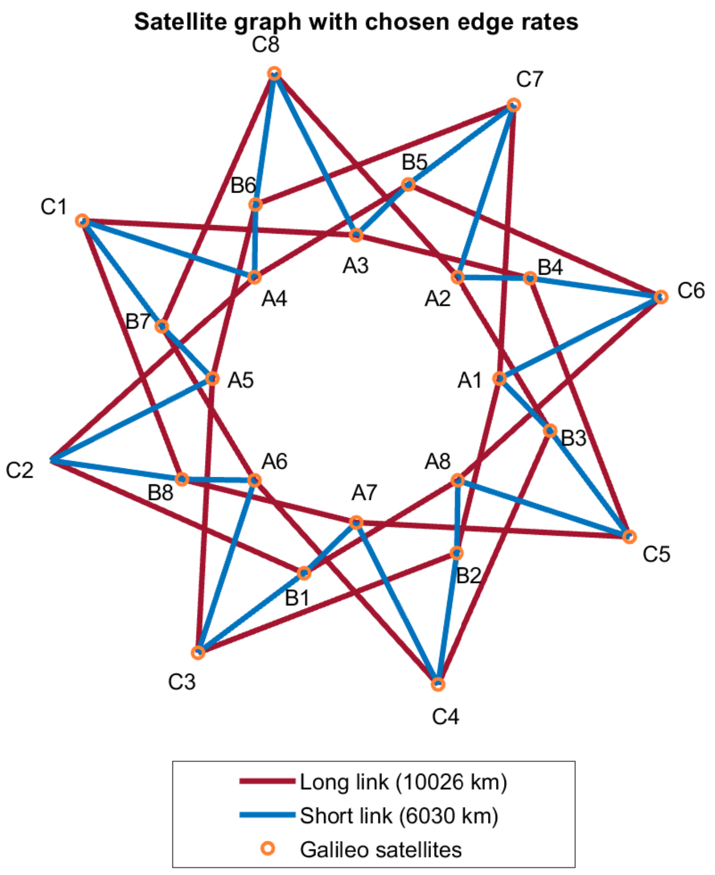


Figure 6.2: Graph representation of the inter-satellite network spanned by quantum links.

derived in the infinite key length regime. A more realistic model developed in [101] accounts for the effects of finite key length and exploits a more realistic model of the satellite features and geometry in the calculation of the dark count probability. Moreover a heuristic approach is taken in order to optimize the SKR, verifying whether it is more convenient to perform a single QKD session with block length n_x or to divide the time window into multiple shorter QKD sessions. The results of the simulations reveal that, fixed the number of pulses per second, N_x, it is more efficient to perform a single QKD session with a longer block length. Since the feasible time windows for QKD partially overlap, each satellite has one additional degree of freedom, that is the switching time between two QKD sessions with different satellites. An optimization was performed in order to find the switching time that maximizes the sum or the minimum of the SKR in the two quantum links.

More specifically the switching time t_s takes values in a window that spans from the trasmissivity peak of the short link to the trasmissivity peak of the long link, sampled at intervals of 50.24s, for a total window duration of 52.75min. As the constellation is symmetric, all switching times can be derived from a single one and therefore it is sufficient to analyze the switching time between links A1–B3 and A1–B2. Let us define S_{B3-B2} as the *normalized switching time* between the short link and the long link:

$$S_{B3-B2} = \frac{t_s}{T_w}. \tag{6.1}$$

Fig. 6.3 represents the value of $\frac{C_\ell + C_s}{2}$ and $\min_{t_s}\{C_\ell, C_s\}$ for different values of the normalized switchint time.

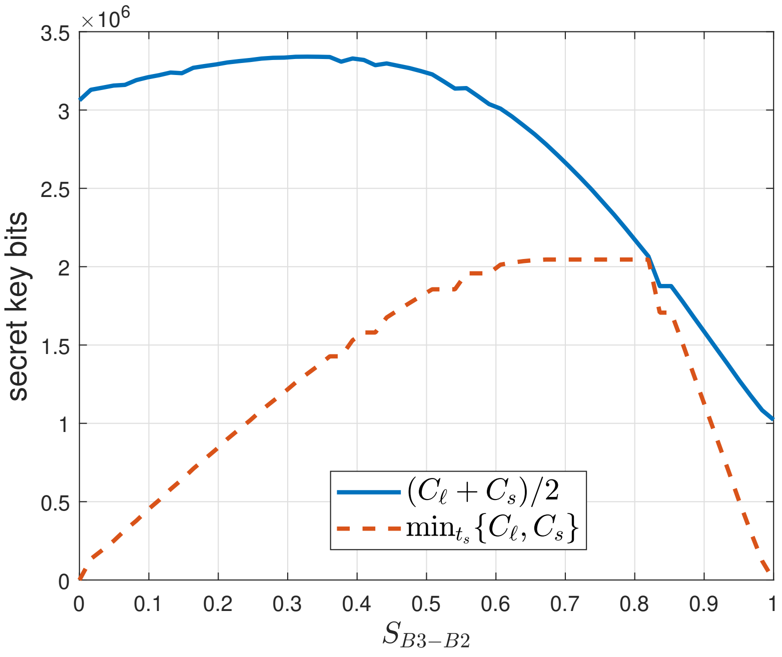


Figure 6.3: SKR for varying S_{B3-B2}.

Regardless of the actual achievable SKR after tuning the optical transmitter and the switching time, the analysis indicate that inter satellite quantum links allow to achieve a reasonable SKR in the order of a few megabit per orbital period for each satellite pair. As there are two types of links, the SKR can be traded-off between the two through the tuning of the switching time between links. The choice of this parameter should be driven by the needs of the key distribution protocol, which will be discussed in the following.

6.1 Inter-satellite key distribution protocol

The target of the key distribution protocol devised in this chapter is to obtain a fully connected satellite network, where each satellite pair (i, j) shares an unconditionally secret key.

The definition of unconditional (or perfect) secrecy as given by Shannon in [102], states that the *a posteriori* probability of ciphertext $c \in \mathcal{C}$ given message $m \in \mathcal{M}$, $P(c|m)$ is equal to the *a priori* probability, $P(c)$, $\forall c \in \mathcal{C}$. This implies that the ciphertext is independent of the message and therefore an attacker that intercepts one or more $c_i,\ i = 1, \dots, n$ can derive no information about $m_i,\ i = -\infty, \dots, +\infty$. This goal can be achieved through the well-known one-time-pad. The drawback of perfect secrecy is that it requires a key with the same entropy as the message. However, since in this scenario the message is constituted by keys, perfect secrecy may not be excessively demanding. Keys can be shared and updated in the network by exploiting the bits established through quantum links (that benefit from the feature of perfect secrecy as well).

Therefore a multi-hop unconditionally secure key distribution protocol can be devised by using one-time-pad through the links established with quantum communications. In order to establish a key between the satellite pair (i, j), non-adjacent in the quantum link graph, a number of bits will be securely generated by i or j and sent to the counterpart through the links of the graph in Fig. 6.2, encrypted with OTP. The secret bits used for encryption can not be used twice and are considered to be wasted. With reference to Fig. 6.2, the SKR of each physical link ($r_{(p,q)}$, $\forall p, q$) represents how many bits per orbital period can be used for OTP. The main targets of the key distribution protocol are identified as the following:

Pairwise key establishment: select one satellite pair (i, j) and maximize the secret key rate $R_{(i,j)}$. Assess the variation of $R^*_{(i,j)}$ for different pairs (i, j) in the network.

Connected graph key establishment: establish a secure connected satellite graph with fairness. Each satellite must share a $m-$bit long secure key with all the others: $R^*_{(i,j)} \geq m,\ \forall (i, j)$, with $m > 0$. Optimize the network by finding the maximum feasible value of m.

We proceed by introducing a new requirement in our key establishment protocol: collusion resistance. Let us relax the security assumptions by allowing two of the satellites to be hacked by Eve, that can now eavesdrop the secret key bits that are distributed through both the hacked satellites.

Threshold secret sharing [19] is a well established cryptographic algorithm devised by Shamir in 1979 that allows to split a secret into n distinct *shares* and then retrieve it by combining k out of n chunks of information. For our purpose a (4,3) threshold secret sharing scheme is suitable: there are always exactly four node-disjoint paths connecting each satellite pair in the quantum satellite link graph. Therefore it is always feasible to deliver four distinct shares of information from node i to node j in such a way that even if two shares are eavesdropped, Eve won't be able to compute the final key. Moreover, this solution allows for some degree of loss-tolerance against channel errors, as one ill received share can be discarded, using the remaining three to retrieve the final key.

We extend the previous problems with the additional constraint of implementing a (4,3) threshold secret sharing protocol:

Pairwise key establishment with secret sharing: select one satellite pair (i, j) and maximize the secret key rate $R_{(i,j)}/4$, subject to the constraint that no node in the network (except i and j) must receive a rate higher than $R_{(i,j)}/4$ and the total rate received by node j must be $R_{(i,j)}$. Assess the variation of $R^*_{(i,j)}/4$ for different pairs (i, j).

Connected graph key establishment with secret sharing: establish a secure connected satellite graph with fairness and secret sharing. Each satellite must share a $m-$bit long secure key with all the others: $R^*_{(i,j)}/4 > m$, $\forall(i, j)$, with $m > 0$. Optimize the network by finding the maximum value of m, subject to the constraint that no node in the network (except i and j) must receive a rate higher than $R_{(i,j)}/4$, $\forall(i, j)$ and the total rate received by node j must be $R_{(i,j)}$.

The input to each optimization problem are the physical SKR (or secret key capacity per orbital period, SKC) of quantum links. The SKCs are of two types, $r_{i,j} = C_l$ for all $e_{i,j} \in S_{\text{long}}$ and $r_{i,j} = C_s$ for all $e_{i,j} \in S_{\text{short}}$, where S_{short} and S_{long} are the sets containing all short links and long links, respectively.

The output comprises both the optimized key distribution protocol with the secret key bits routing across the network and the maximum achievable secret key rate per inter-satellite link. By iterating the optimization for different values of the switching time between quantum links it is possible to assess which value of this parameter allows for a higher SKR, thus allowing to optimize the quantum link scheduling.

6.1.1 Pairwise key establishment

The aim of this optimization is to assess how many secret key bits can be shared per orbital period between any pair of nodes by exploiting the full physical link capacity. This problem refers to the situation in which a satellite pair suddenly needs to renew its shared key for any reason (e.g., key expiration or compromise). We are therefore assessing the best strategy to perform this re-keying in the ideal case where all the physical links are dedicated to this pairwise key establishment.

Let us call $x_{k,\ell}$ the SKR on edge $e_{k,\ell}$, and $\mathbf{x} = \{x_{i,j} : e_{i,j} \in \mathcal{G}\}$. For each pair (i, j), we are interested in solving:

$$t^* = \max_{\mathbf{x}} \sum_{\ell} x_{i,\ell}, \text{ or equivalently, } t^* = \max_{\mathbf{x}} \sum_{k} x_{k,j}. \tag{6.2}$$

under the following constraints:

$$\begin{cases} x_{k,\ell} \geq 0 \;\; \forall k, \ell : e_{k,\ell} \in \mathcal{G} & \text{flow positivity} \\ x_{k,\ell} \leq r_{k,\ell} \;\; \forall k, \ell : e_{k,\ell} \in \mathcal{G} & \text{capacity} \\ \sum_k x_{k,n} = \sum_\ell x_{n,\ell}, \;\; \forall n \neq i, j & \text{flow conservation} \end{cases}$$

From Fig. 6.2 it can be seen that the maximum expected value for the result is

$$\max_{\mathbf{x}} \sum_{\ell} x_{i,\ell} \leq 2C_\ell + 2C_s \tag{6.3}$$

that is the sum of the SKC of all incident edges for any node in the network.

The problem can be extended to become resilient to man in the middle attacks from maximum one node in the network and secret against the eavesdropping from two hacked nodes in the network. For this purpose we select Shamir's polynomial $(4, 3)$ threshold secret sharing. We must ensure that for each satellite pair (i, j) 4 shares of secret bits of length t are distributed on distinct and disjoint paths (i.e., none of the paths must share edges or nodes with the others). This guarantees that no node sees more than one share, except for i and j. The final key will be obtained by combining the shares (at least three), giving a t bit long unconditionally secure key. The additional resilience comes at the cost of an increased number of secret key bits to distribute, that will lower the SKR.

The problem is stated similarly as the previous one:

$$t^* = \max_{\mathbf{x}} t = \max_{\mathbf{x}} \frac{1}{4} \sum_{\ell} x_{i,\ell}, \tag{6.4}$$

$$\text{or,} \;\; \max_{\mathbf{x}} \frac{1}{4} \sum_{k} x_{k,j}. \tag{6.5}$$

with the addition of the following constraints:

$$\begin{cases} \dots \\ \sum_k x_{k,n} \leq t, \;\; \forall n \neq i, j & \text{secrecy} \\ \sum_\ell x_{i,\ell} \geq 4t, \;\; \forall n \neq i, j & \text{correctness} \end{cases}$$

Notice that this problem formulation forces secret sharing only among satellites that are not adjacent in the quantum link graph: in the adjacent case, indeed, the quantum key bits can be directly used as a key. Four disjoint paths connecting each node to any other always exist and at least two of them contain one link with lowest capacity. Since we are enforcing that all shares must have the same length, the limiting factor is indeed the lowest capacity between the two link types.

Results

As expected, the solution to the two optimization problems is upper bounded by $2C_\ell + 2C_s$ in the first instance and by $\min(C_\ell, C_s)$ in the second. The solution was computed for multiple satellite pairs and over several values of S_{B3-B2}, the normalized switching time. The solution to the latter problem always coincides with the upper bound, $\min(C_\ell, C_s)$.

In the former problem the result shows that the upper bound is reached for all satellite pairs for $S_{B3-B2} < 0.98$. For $S_{B3-B2} > 0.98$ the two C_ℓ and C_s are too unbalanced and the upper bound for the SKR is reached only for a few satellite pairs. The "privileged" nodes for which the SKR reaches the upper bound are coincidentally those from which two disjoint paths can be found that reach the source node by only traversing long links (notice that in this case $C_\ell \geq C_s$). As an example for $i =$ A1 the "privileged" nodes are B2, C3, A5, B6 and C7.

The switching time can be optimized for any of these two problems in order to maximize the SKR for *any node pair*. As expected, the solution behaves as $S^*_{B3-B2} = \arg\max_{S_{B3-B2}} (C_\ell + C_s)$ for the first problem and $S^*_{B3-B2} = \arg\max_{S_{B3-B2}} (\min(C_\ell, C_s))$ for the second problem, following the curves represented in Fig. 6.3.

Fig. 6.4 represents the routing solution for satellite pair A1-A5 in problem 1.

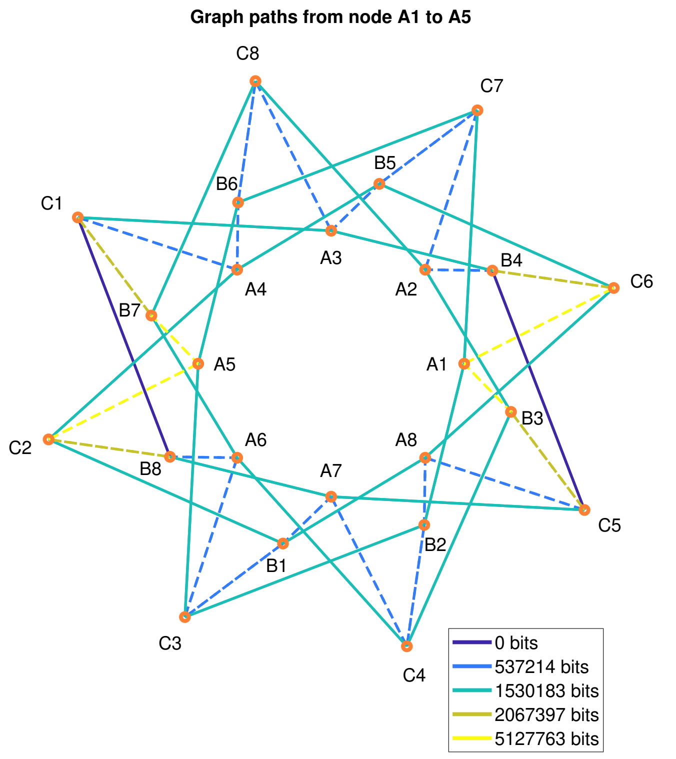


Figure 6.4: Example of key agreement protocol between nodes A1 and A5 by exploiting the full network capacity.

6.1.2 Connected graph key establishment

The key agreement problem is now extended to the whole satellite network. We are interested in optimizing the key establishment protocol in order to create a secure connected graph, that is, we want each SV i to share a key of length $R_{i,j}$ with each other SV j. In order to uniformly distribute the key material over the satellite network, we optimize the protocol by looking for:

$$t^* = \max\{t\} : R^{i,j} \geq t, \ \forall i, j \tag{6.6}$$

The solution to this problem provides an algorithm to perform a re-keying of the whole satellite network in such a way that all quantum secure bits are exploited with maximum efficiency, securely connecting each satellite pair with a t^*-bit symmetric key. Each link can be used in both directions to relay secure key bits for any pair of satellites. Let us call $x_{k,\ell}^{i,j}$ the number of bits of key $K^{i,j}$ that are relayed through the link $(k \to \ell)$. In the flow optimization problem we use a directed graph where each edge is split into two directed edges, since both directions are feasible for each satellite link. Therefore we differentiate between key bits $x_{k,\ell}^{i,j}$, generated by i and forwarded to j and $x_{k,\ell}^{j,i}$, generated by j and forwarded to i. Both $x_{k,\ell}^{i,j}$ and $x_{k,\ell}^{j,i}$ will be used for the key established between i and j, whose total length will be:

$$R^{i,j} = \sum_{\ell} x_{i,\ell}^{i,j} + \sum_{\ell} x_{j,\ell}^{j,i} \tag{6.7}$$

so the problem takes the form:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} t \tag{6.8}$$

subject to the following constraints:

$$\begin{cases} x_{k,\ell}^{i,j} \geq 0 \ \forall k, \ell, i, j & \text{flow positivity} \\ \sum_{i,j} x_{k,\ell}^{i,j} \leq r_{k,\ell} \ \forall k, \ell : e_{k,\ell} \in \mathcal{G} & \text{capacity} \\ \sum_k x_{k,n}^{i,j} = \sum_\ell x_{n,\ell}^{i,j}, \ \forall n \neq i, j & \text{flow conserv.} \\ R^{i,j} = \sum_\ell x_{i,\ell}^{i,j} + \sum_\ell x_{j,\ell}^{j,i} \geq t \ \forall i, j & \text{minimum flow} \end{cases}$$

The output of the optimization provides the optimal routing for the key agreement between each satellite pair. The result depends on the values of C_l and C_s, that is on the time of switch from one satellite to the other. This parameter influences both the routing algorithm and the actual value of t^*, the number of secure key bits established between each satellite pair. To analyze this dependence we solved the optimization problem for all possible values of S_{B3-B2}, and the result is represented in figure 6.5.

As can be seen from the figure, the number of the unconditionally secure bits that each satellite pair can share as a function of the switching time is a concave function. The maximum is obtained for $S_{B3-B2} = 0.6$. Notice that this does not coincide with $\max(C_l + C_s)$, which corresponds to $S_{B3-B2} = 0.4$. The reason may lie in the fact that the two subgraphs $\mathcal{G}_s$ and $\mathcal{G}_\ell$ spanned by short links and long links respectively have different topological properties. In $\mathcal{G}_s$ indeed a Hamiltonian path exists, whereas none exist in $\mathcal{G}_\ell$. This is a reasonable explanation why maximizing the sum $(C_l + C_s)$ does not lead to the best results in terms of shared secret key bits.

The figure also highlights the efficiency of the key agreement protocol, intended as the ratio between established secure bits and the total number of bits employed for the one-time-pad key establishment. It can be noticed that the maximum efficiency does not coincide with the maximum t^*, peaking at $S_{B3-B2} = 0.88$.

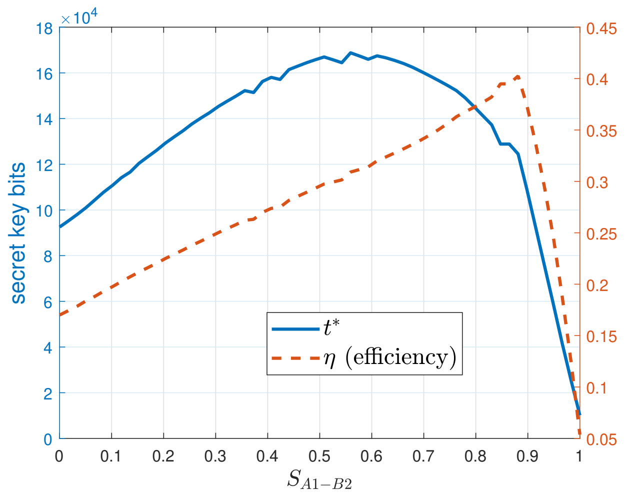


Figure 6.5: t^* for varying t_s and the efficiency η of the key agreement algorithm in terms of the ratio between the total number of exchanged secret key bits and the total number of quantum distributed bits.

The routing algorithm outputs the links that each satellite has to exploit to establish t^* secure bits with another satellite. Each link costs t^* of the bits that were collected through the quantum key agreement protocol, since one-time-pad needs to be performed in order to preserve unconditional security. It is possible to observe that in some cases the t^* bits are distributed altogether trough the same path, while in some others more than one path is used, as shown in Fig. 6.6.

As seen in Sec. 6.1.1, the optimization problem can be extended by implementing a (4,3) threshold secret sharing. The aim is that of devising a key agreement protocol capable of obtaining a shared key between each satellite pair, with some level of robustness against man in the middle attacks and eavesdropping. The secret sharing constraints are included in the problem formulation in order to avoid that any of the relay nodes sees more than one out of four shared of the final key (secrecy), ensuring that the final number of shares for each satellite pair is four (correctness).

$$\begin{cases} R^{i,j} = \sum_\ell x^{i,j}_{i,\ell} + \sum_\ell x^{j,i}_{j,\ell} \geq 4t, \quad \forall i,j & \text{correctness} \\ \sum_k (x^{i,j}_{k,n} + x^{j,i}_{k,n}) \leq t, \quad \forall n \neq i,j & \text{secrecy} \end{cases}$$

Notice that secret sharing is imposed only between satellites that are not adjacent in the quantum link network: in that case, indeed, the shared quantum key bits can be directly used as a key. The solution to this optimization problem will provide the maximum length of the established keys as well as the routing protocol that allows to perform the key agreement according to the secret sharing constraints. Again the solution depends on the time of switch in the quantum key distribution protocol and therefore the problem was solved for various values of S_{B3-B2}. An example of the routing output of this optimization problem is given in Fig. 6.7

Fig. 6.8 represents t^*, the maximum number of secret key bits that can be distributed among each couple of satellites, thus connecting all the network, with the secret sharing requirements. The efficiency is also showed, representing the ratio between established

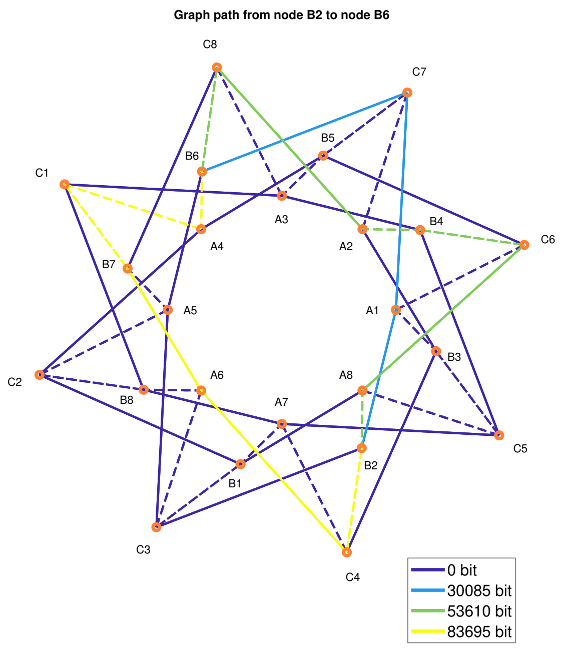


Figure 6.6: Example result of the routing optimization for unconditionally secure key agreement between two satellites. This result has been obtained by setting $S_{B3-B2} = 0.6$.

secure bits and the total number of bits wasted for the one-time-pad key establishment. It can be noticed that the behavior of the two metrics is very similar to that of the previous optimization problem. The number of secret key bits is now lower than one quarter with respect to the solution without secret sharing. As discussed, this loss is expected, since four shares of L bits are now combined in order to give a single L bit key, but interestingly the cost of these additional security features does not just scale of a factor four and is instead a bit higher. The behavior of t^* seems to depend both on the sum $C_\ell + C_s$ and on the minimum of the two, as it can be deduced by comparison with Fig. 6.3.

6.2 Conclusions

The analysis carried out in this chapter achieves two main goals: an algorithm is devised to efficiently perform key distribution over the whole satellite network, while providing some insight on the best choice for the time of switch between short and long inter-satellite quantum links.

- The result of Section 6.1.1 suggests that in order to maximize the number of secure bits established between a specific satellite pair, the optimal switching time is the one that maximizes *the sum of the link capacities*, $t_s^* = \arg\max_{t_s}(C_l + C_s)$.
- When secret sharing is introduced to the pairwise key agreement problem, the optimal switching time is the one that maximizes the *minimum link capacity*, $t_s^* = \arg\max_{t_s}(\min(C_\ell, C_s))$.
- Section 6.1.2 deals with a key agreement protocol for establishing a connected graph.

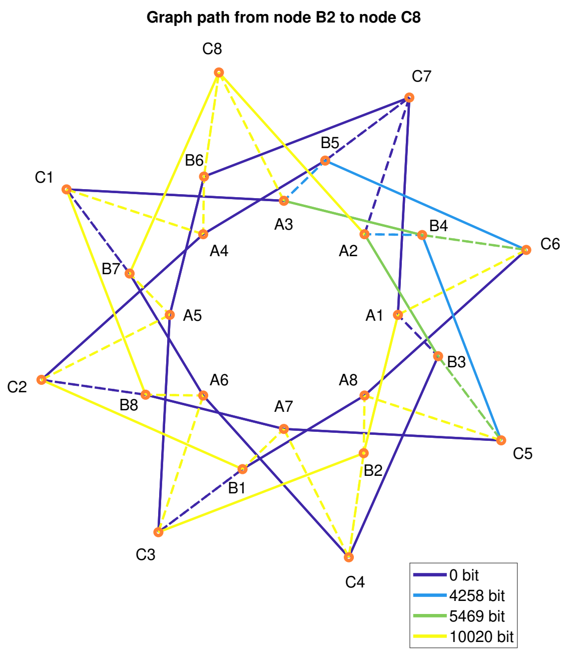


Figure 6.7: Example result of the routing optimization for unconditionally secure key agreement among the whole network. This result has been obtained by setting $S_{B3-B2} = 0.33$.

The results show that regardless of the presence of the secret sharing feature, the optimal switching time is in between the two previous results, underlining the dependence on both the sum and the minimum among link capacities.

Apart from these considerations on the optimal switching time, evidence is provided that an efficient key agreement protocol can be devised to perform unconditionally secure key agreement across the whole satellite network. The provided methodology allows to derive the optimal routing that maximizes the number of established secure bits while minimizing the quantum agreed bit consumption. The devised algorithm allows to formulate each key agreement scenario as a minimum cost flow in the network graph, computing the optimal solution for each value of the link capacities.

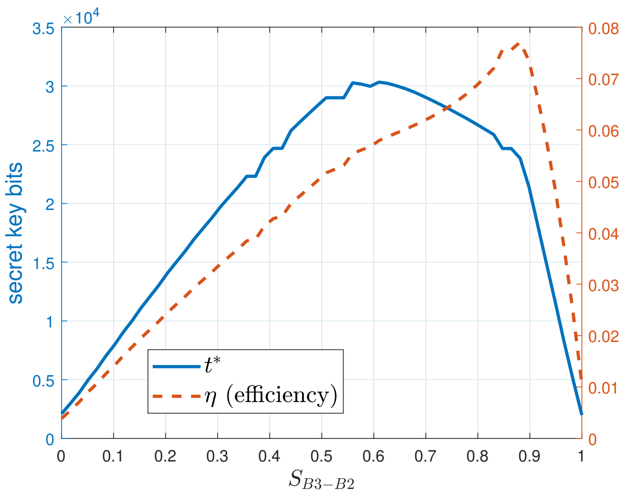


Figure 6.8: t^* for the key establishment optimization on the whole graph with secret sharing. The result is expressed for varying t_s. The efficiency η of the key agreement algorithm is also represented, in terms of the ratio between the obtained secret key bits and the quantum distributed bits.

Chapter 7

Anti-spoofing in low-grade smartphone applications

The security issues in GNSS have been widely investigated in the literature and several anti-spoofing mechanisms have been proposed for implementation. These can either operate at the system level, with the insertion of security features into the signal in space (SIS), or at the receiver level, with the development of robust signal processing techniques. Most of the investigated solutions, however, were developed for a medium to high grade receiver as the target consumer device. With the widespread of location based services, positioning capabilities have been integrated on all sorts of low grade devices. For this reason the spoofing threat is becoming a concern even in low grade scenarios such as mobile handsets. In recent years indeed the variety of smartphone-based applications for user and society critical tasks has vastly increased:

- Smartphone payment systems are now used by many customers and card fraud is a well known issue. Indeed, credit card companies currently look for inconsistencies between the site of a transaction and the estimated location of the cardholder, raising warnings in case of a mismatch.
- Several companies use tracking applications installed on smartphones or tablets provided to their employees for workforce management.
- Enhanced 911 (E911) is a standard for emergency calls that requires the carrier to deliver the user's positioning information whenever a 911 call takes place. The standard also defines specific requirements for the positioning accuracy.
- Crowdsourcing services often include geographical coordinates in the information collected from the users.
- Some security-critical software applications rely on GNSS positioning. For instance Google *Trusted Places* allows the user to unlock the mobile device without any kind of authentication (passwords, fingerprints, etc..) in case the phone is located in specific positions.

The above considerations motivate a study on GNSS positioning resilience in smartphones and how to improve it with the current handset capabilities. Moreover, a recent spoofing incident happened at ION GNSS+ 2017, causing a number of devices to be accidentally spoofed by a GNSS simulator with non-properly terminated RF connectors, [103]. This event suggests that smartphones are not immune to spoofing, and it has further encouraged our investigation on the entity of this threat. Resource constrained mobile phones are designed to provide position, velocity and time to the end user even in the most challenging environments, where the GNSS signal is hardly available in the first place. This approach pursues usability and energy efficiency rather than security, thus leaving the smartphones exposed to potential vulnerabilities. Classic anti-spoofing solution are hardly applicable in this scenario, as they are in general too demanding in terms of cost or energy requirements. As an example, the work presented in [104] investigates a promising receiver autonomous anti-spoofing mechanism based on carrier phase difference for multi-antenna receivers. The technique of vestigial signal defense [105] is another receiver autonomous technique that monitors the correlation function to detect spoofing events. In hand-held devices, however, such techniques are not applicable either because they require additional hardware [104] or due to their computational cost [105].

While autonomous receivers solely rely on GNSS to calculate their position and time, mobile handsets benefit from other options. Other sources, such as signals of opportunity (WiFi, cellular networks, Bluetooth) or the on board sensors (accelerometer, digital compass, clock, etc.) provide useful redundancy that is currently exploited to improve the efficiency and availability of positioning services. (e.g., indoor positioning, dead reckoning in tunnels, etc.). A-GNSS has contributed to the widespread of GNSS positioning in handsets, allowing mobile devices to retrieve system information (ephemeris data, frequency and code delay estimates, etc.) from an aiding channel. These resources have greatly improved the positioning performance of GNSS based applications, allowing mobile devices to provide a PVT solution even in the most challenging conditions. Since GNSS spoofing is now an emerging issue, there is no reason why these beneficial sources should not be used as an ally for security purposes as well.

The aim of this work, which was published in [106], is to propose directions for improving the resilience of mass market GNSS modules against spoofing attacks. Some simple consistency checks would indeed increase the user awareness, minimizing the impact on the user segment without requiring modifications to the ground or space segment. With this purpose, we devised an experimental campaign to analyze how mobile devices behave when exposed to a spoofing GNSS signal that is inconsistent with any other positioning sources. The experiments show that in the current chipsets generation these consistency checks are either not implemented or at least that the resulting inconsistencies are not reported to the user.

Even though GNSS spoofing on handsets is a known issue, no previous work has investigated it extensively. In [107] a few devices were tested, but only under jamming attacks. In [108] the design of a low cost GNSS simulator was discussed and some tests were performed on a couple of smartphones, exposing them to a time and location spoofing attack. The tests were limited in the number of devices and in the type of trials, nevertheless they highlighted the vulnerabilities we want to better investigate in this work. The authors of [109] also experimented with spoofing on mobile devices, stating that forcing a fake position fix on smartphones is relatively easy. They showed a successful position fix on an iPhone 6, but they observed that spoofing an Acer android phone was harder, claiming that the smartphone used integrated side information from WiFi and cellular networks in the positioning protocol. In fact, on our devices we never reported such resistance to spoofing attacks. Neither [109] nor [110] performed a systematic experimental campaign on smartphones, since their focus was mainly on professional GNSS receivers. Several results on spoofing detection techniques have been presented in the literature, targeting more sophisticated receivers and leveraging different features of the spoofing signal. Some works focused on CNR monitoring for spoofing detection [111]; others propose a detection method based on the comparison between IMU measurements and the computed PVT [112]. Mitigation techniques have also been investigated, as those in [113], specific for multi-antenna receivers. Most of these techniques, when applied to mobile phones, would require substantial changes either in the receiver logic (leading to the use of more expensive chipsets) or to the SIS (as for cryptographic anti spoofing techniques, [4,8,11]). This work, on the other hand, focuses on simple counter-measures that are software based, and hence feasible even for the current generation of mobile devices.

7.1 Positioning information sources

There are several sources of positioning information available to mobile devices. Let us summarize in the following the ones we considered in this work.

Navigation Message

The GNSS navigation message contains all the information necessary to compute the PVT solution and is modulated on the SIS. These are the ephemeris data, the time reference, time corrections, the ionospheric corrections, and the almanac data, that have lower accuracy but longer validity. Let us consider in particular the GPS navigation message, represented in Fig. 7.1, that is divided into subframes and takes approximately 12 minutes.

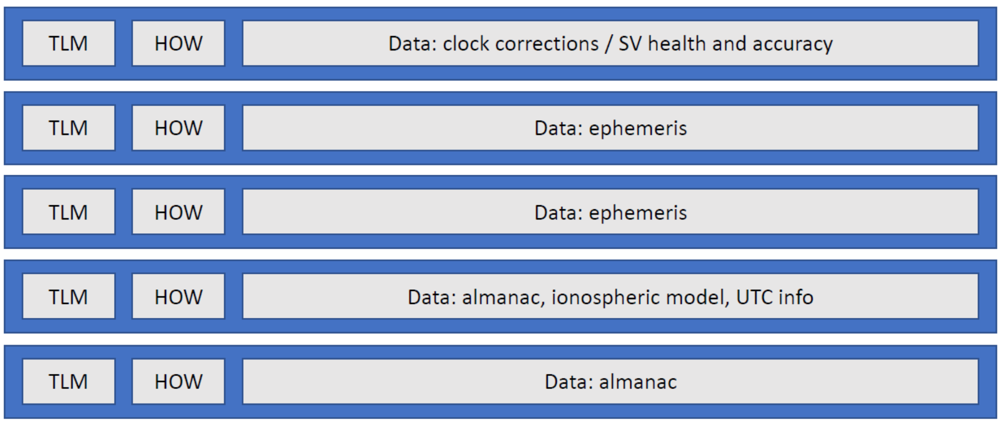

TLM
HOW
Data: clock corrections / SV health and accuracy
TLM
HOW
Data: ephemeris
TLM
HOW
Data: ephemeris
TLM
HOW
Data: almanac, ionospheric model, UTC info
TLM
HOW
Data: almanac

Figure 7.1: Structure of the GPS navigation message frame. Each of the five subframes lasts 60 s, which is the length of the horizontal time axis.

The telemetry word is the first word of every subframe and begins with an 8-bit preamble for synchronization purposes. The handover word is the second word of each subframe, and contains the 17 most significant bits of the GPS time of week (TOW) relative to the beginning of the next subframe. If in general most of the other data can be downloaded from other sources (e.g., with A-GNSS), these two words contain the information necessary for synchronizing and obtaining the transmission time of the signal (that can also be found with higher precision in another field of the navigation message).

Mobile Networks

Handsets are most likely to be connected to a mobile data network, and thus to a specific serving base station which can provide a time and position reference, according to the network type and the exploited protocol.

As reported in [114], all cellular networks have intrinsic location capabilities that can be exploited by location based services. The GSM/GPRS/EDGE location services (LCS) architecture has several standardized positioning methods mostly based on time difference of arrivals (e.g., multilateration). These allow the user to obtain an estimate of either its own position or the location of the serving network entity. With UMTS networks, several other positioning mechanisms were defined, as cell-ID positioning, angles of arrival (multiangulation), or almanac-based DGPS, improving positioning resolution. LTE LCS architectures further improved positioning protocols, e.g., with the introduction of specific positioning reference signals, designed solely for localization purposes, improving availability and precision. As claimed in [115], even position estimates derived from cellular networks may be subject to spoofing attacks. The attacker could indeed tamper with the estimate by either impersonating remote infrastructure or by tampering with the service database. Thus it is worth remarking that exploited sources of position redundancy need to be authenticated.

Assisted GNSS

A-GNSS provides mobile receivers with assistance information coming from alternative channels that improves positioning performance, e.g., by reducing the Time To First Fix (TTFF). An alternative channel can be an Internet connection that, by transmitting at a higher rate, provides the receiver a position fix *before* having to download the whole navigation message from the SIS. Depending on the specific A-GNSS implementation the assistance information may comprise: ephemeris (and/or almanac), approximate a-priori user position, estimated Doppler frequency, estimated code-delay and a time reference, as detailed in [116, Ch. 3]. There are several implementations of A-GNSS and two main standardized frameworks [117]: control plane and user plane A-GNSS. Control plane protocols directly rely on the signaling layers of the communication networks. Examples of these protocols are the radio resource location services protocol (RRLP) [118] and the radio resource control (RRC) [119] protocol for UMTS, both defined by 3GPP. The RRLC protocol is specific for LCS in GSM networks, and it uses no authentication; RRC is a UMTS protocol. 3GPP also defines a LCS protocol for LTE networks, that supports both user plane and control plane positioning.

User plane protocols are based on the application layer, even though packets still exploit data formats defined by control plane protocols. These are mainly defined by the open mobile alliance (OMA), e.g., the OMA secure user plain location (OMA-SUPL) protocol [120]. While the 3GPP standard exploits cellular networks, OMA-SUPL can be implemented in a wide variety of contexts (e.g., mobile network or open internet/WLAN environments). GSM, only supports user authentication while UMTS and LTE employ mutual authentication between user and network. WiFi connections also support mutual authentication (e.g., through EAP security protocols). Therefore the information acquired through A-GNSS can in general be considered secure and trustworthy.

Device Motion Sensors

Nearly all smartphones today integrate a set of measurements taken from sensors such as motion (accelerometers), rotation (gyroscopes) and magnetic field (magnetometers) sensors. The data gathered by each sensor alone are of little help in estimating the device's absolute position, but by processing and integrating all information (possibly also with GNSS measurements) an estimate can be obtained. Sensor fusion has been so far employed for navigation aiding purposes such as pedestrian dead reckoning [121] and indoor navigation [122], [123], [124].

7.2 Exploiting Side Information For Secure Positioning

This section proposes to integrate the above mentioned side information in the positioning algorithm for security purposes. None of the reviewed GNSS positioning applications available in the play store, indeed, offers a spoofing detection indicator. Since smartphones are meant to be energy efficient, using computationally expensive tools for anti-spoofing is unrealistic. For this reason the focus is on software-based cross checks that allow to improve spoofing resilience at a minimum cost.

The model for spoofing attacks considered in this work consists in the transmission of a GNSS signal over-the-air. The attacker has complete control on the features of the signal, such as the power, the intended time and coordinates, the navigation message, etc. The attacker is also assumed to be able to suppress the authentic GNSS signal e.g., by transmitting a spoofing signal with higher power.

In the following four steps are proposed to improve the smartphones resilience to spoofing attacks. These expedients are not sufficient to achieve a complete PVT robustness, as they may not protect against more sophisticated spoofing attacks. However they have the advantage of being easy to implement even in today's receivers, and could be later complemented by the next generations of GNSS signals that may allow for more effective authentication techniques.

7.2.1 Position History

Trivial location spoofing attacks, where the attacker simply forces a sudden change of coordinates, are easy to detect, as they exhibit a significant displacement in a relatively short time. By keeping a history of the last computed position, even when other connections are not available, the device could verify that such a sudden change is not physically feasible, and the attack could be detected. A finer approach could monitor other observables as well, such as sudden jumps of the C/N_0. However in a cellular scenario such jumps can be observed even in authentic conditions, caused by sudden shadowing due to the diverse urban environment. Moreover, since the aim of this investigation is to point out the most readily available resilience enhancements, it makes sense to limit testing to more immediate and less ambiguous information.

7.2.2 Cross-check of Time and Position

Another means for detecting an inconsistency comes from the availability of network connections and corresponding geographical information. For instance, geolocation of a WiFi access point is possible by its SSID and MAC address, and analogously by the ID of detected cells in a mobile network.

In devising anti-spoofing mechanisms based on side channels, one should consider that network connections may themselves be vulnerable to spoofing. For instance, spoofing the SSID of a WiFi network is easy, and several websites exist that can map an SSID to a specific location. Therefore localization aiding by means of SSID sensing should not be trusted as a source of authentic positioning information. On the other hand, when devices are connected to a *trusted* WiFi network, UMTS or LTE base station (i.e., having performed mutual authentication), this information can be considered trustworthy, and used to identify an approximate area inside which the device is located with a certain level of trust.

On the contrary GSM only authenticates the users, and the spoofing of a GSM base station is quite feasible. Under these considerations the available side information should be weighted according to the security level of the connection it comes from.

Another trusted source of information comes from the aiding data obtained through A-GNSS. Since all implementations are different, the available information may not be the same for every device, but according to [116] smartphones using this service may benefit from:

- the location of a reference entity (provided that the service is delivered through cellular networks);
- the satellites in view in the area;
- some information to reduce the search space in acquisition phase;
- the ephemeris data;
- a time reference.

Any of these authenticated data could be used to enhance security and restrict the freedom of an attacker. The information related to an approximate estimate of the position can be verified after obtaining the PVT solution, checking that the result falls inside the provided estimated area. As an example, if the satellites in view are completely different from those reported in the A-GNSS information, the positioning protocol could be suspended and an alert could be raised.

Cross checking the GNSS signal with authenticated time information can also protect the devices from time spoofing attacks, by alerting the application layer when the GPS signal time presents discrepancies with an external authenticated time source. This procedure would allow to avoid dangerous sudden time corrections in the far past or future, that could result in a malfunctioning device.

7.2.3 Compliance of the Navigation Message

A-GNSS allows to retrieve the content of the navigation message, eliminating the need for demodulation of the SIS. Nonetheless, in order to be resilient against spoofing, it is important to periodically check the coherence of the navigation message content with the available authenticated data. The receiver should verify that the structure of the navigation message is well formed (e.g., frame format and parameter fields).

For most of the ephemeris parameters the range of values directly derive from their bit representation. However, some of these parameters have a different validity range, that is specified in [125]. As an example, the orbit eccentricity values span from 0 to 0.03, and therefore all representable values outside this range are not valid. As a matter of fact, with the 32-bit field reserved for the eccentricity, with the defined scale factor of 2^{-33}, the maximum representable value is 0.5.

It is good practice to validate the compliance of data to the standard defined in the GPS Iterface Control Document (ICD), making sure their value lies inside the specified validity range. The benefit of this validation is twofold. Indeed if an invalid message were accepted not only it would corrupt the current position calculation but it would also prevent the receiver from attempting to dowloand the correct data once the authentic signal becomes available again. The latter situation could impair the positioning capabilities of the device for a substantial amount of time.

Therefore we suggest to continuously check the consistency, compliance and correctness of the navigation message as a best practice. The periodicity of this cross-check shall be defined according to a trade off between energy efficiency requirements and security.

7.2.4 Cross-Check With the Motion Sensors

According to the considered spoofing model, nothing prevents an attacker from spoofing a fake *trajectory*. Detecting this attack by using side information is not a trivial problem, as it requires telling two different trajectories apart from one another. In this paper we propose to use sensor data for integrity checking, with the aim of detecting potential incoherency between the GNSS PVT solution and the data gathered from accelerometer, magnetometer and gyroscope. Unfortunately the low quality of the devices sensors and the moving reference system on which the measurements are performed makes the processing of these data complicated. However, when the motion implied by the spoofed trajectory is dramatically different from the authentic asset of the phone (e.g., static vs moving) an immediate help can come from the sensor, as we will show in Sec. 7.3.6.

7.3 Experimental Campaign

We carried out an experimental campaign to see whether any of the solutions proposed in section Sec. 7.2 were already implemented, allowing the user to be aware of a possible ongoing attack. In this section we describe the experiments in detail.

Since none of the available navigation apps gives an insight on the performed security checks (e.g., by showing a spoofing flag), we consider the attack *successful* in case the device computes the fake PVT. Although this event may not imply that the device did not perform the suggested security operations, at least it indicates either that the results of such operations were negative (i.e., the attack was undetected) or that the alert they raised was suppressed in favor of the service availability.

Even though usability is an important principle in mobile devices, we claim that smartphones should at least notify the application layer about any detected anomaly that could potentially be linked to an attack. In parallel, a PVT can still be provided, but the task of deciding whether to trust the computed position will be left to the user, that can better evaluate the risk.

7.3.1 Experimental Set-up

Crowd sourcing allowed us to perform an extensive test campaign: a list of the tested smartphones makes and models is reported in Tab. 7.1.

A GNSS software simulator in C++ is the experiment core as it generates the GPS signals for any selected position and time. A bladeRF software-defined radio (SDR) from Nuand [40] is used to modulate the signal, while a u-blox M8T receiver serves as a benchmark, and is connected to the SDR through an SMA cable. The whole setup is showed in Fig. 7.2.

The circuitry has a limited signal leakage, allowing us to receive the artificial signal through any mobile device placed within 1 m distance from the cable without disrupting the behaviour of devices not under test. As the experiment is performed indoor, the legitimate GNSS signal is strongly attenuated. Although this setting may not reflect a realistic spoofing model, it aims at minimizing the possible interference in the surrounding area, and can be seen as a pessimistic scenario in which only the spoofing signal is visible. For the sake of validating our experiments also in the presence of the authentic signal, we performed some tests in an open space, still using RF cables, and we verified that the results did not change, provided that the received power of the spoofing signal is high enough. We limited our trials to GPS, since some of the devices do not support the Galileo constellation yet. The simulator is used in real time and it reproduces the authentic GPS signal for a given geographic coordinate at the exact time in which it is started.

The setup provides complete control over the following aspects of the experiment scenario:

Navigation Message The simulator allows to modify the modulated data by selectively replacing some fields of the navigation message with forged or dummy values.

GPS constellation and receiver coordinates We can select the position and time of the signal reception, and even act on the parameters of the constellation itself (e.g., orbit eccentricity), modifying the transmitted signal accordingly.

Devices Status We have control on the connectivity of the smartphones to data links: WiFi, cellular network and Bluetooth. The initial conditions (e.g., stored aiding data,

Table 7.1: List of Devices

Brand	***Model***	***Year***	***GNSS chipset / SOC***
Apple	iPhone 5	2012	Qualcomm RTR8600
Apple	iPhone 6S	2015	Qualcomm WTR3925
Apple	iPhone SE	2016	Qualcomm MDM9625
Asus	Nexus 7	2013	Qualcomm APQ8064
Asus	Zefone 2	2016	Broadcom BCM47531A1
Google	Pixel	2016	Qualcomm Snapdragon 821
HTC	HTC one M9	2015	Qualcomm Snapdragon 810
Huawei	Honor 8	2016	Broadcom BCM4774
Huawei	Honor 9	2017	Broadcom BCM47531A1
Huawei	p8 lite	2017	HiSilicon Kirin 655
Huawei	p10 lite	2017	HiSilicon Kirin 658
LG	Nexus 5	2013	Qualcomm WTR1605L
LG	Nexus 5x	2015	Qualcomm WTR3925
LG	G6	2017	Qualcomm Snapdragon 835
LG	G3	2015	Qualcomm WTR1625L
Motorola	Moto G (2013) 5	2013	Qualcomm MSM8x26
OnePlus	OnePlus 2	2015	Qualcomm MSM8994
OnePlus	OnePlus 5	2017	Qualcomm Snapdragon 835
Samsung	Galaxy S6 Edge	2015	Broadcom BCM4773
Samsung	Galaxy S6	2015	Samsung Exynos 7420
Samsung	Galaxy S7 Edge	2016	Qualcomm WTR3925
Xiaomi	Mi 4c	2015	Qualcomm Snapdragon 808
Xiaomi	Mi5	2016	Qualcomm WTR3925
Xiaomi	Mi6	2017	Qualcomm Snapdragon 835
Xiaomi	Redmi Note 4x	2017	Qualcomm Snapdragon 625

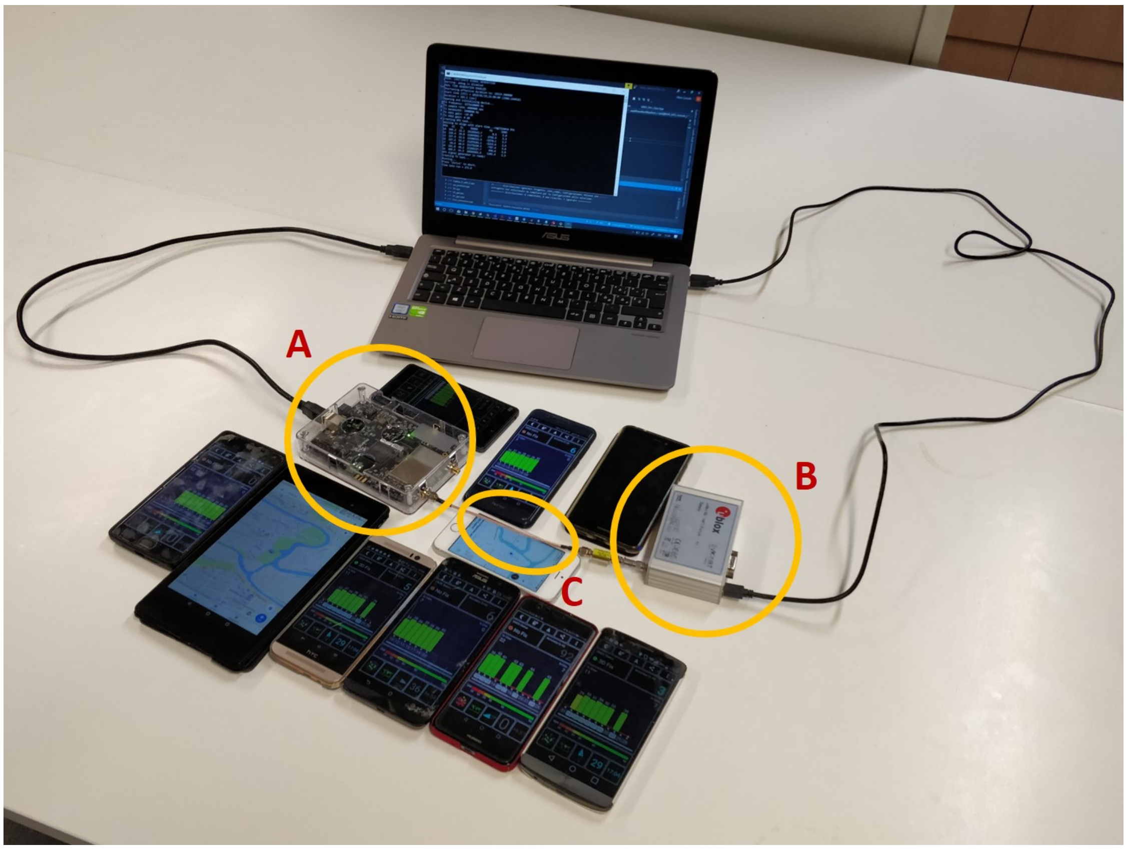


Figure 7.2: A picture of the experiment set-up. In the laptop the GNSS simulator is running and sending information to the SDR (A) which modulates the signal and sends it to the u-blox (B) through the RF cable (C). Around there is a subset of devices under test, with *GPS Test* and *Google Maps* running.

last stored position, etc.), however, are harder to control on high level APIs and the device status might depend on the specific implementation, on which no public information is usually available. In order to minimize the impact of different initial conditions, the same setup sequence has been performed.

During the experiments the *GPS Test* application [126], freely available for Android devices, was used as a tool to inspect the devices response. An experiment is declared successful as soon as the GPS Test application declares a completed 3D fix on the selected spoofed position. We verified that *Google Maps* and GPS Test present the same behavior for what concerns position fixes. On non Android devices (Apple iPhone), instead, the experiment was considered successful when *Apple maps* or Google Maps ended up reporting the selected position. In any case we kept the simulator running for no more than five minutes for each experiment.

In order to assure reproducibility, the same procedure was repeated on every device before starting any of the experiments. This consists in wiping A-GNSS assistance data using the *GPS Test* interface while the device is in airplane mode. To the best of authors' knowledge iOS devices do not have public APIs that support the wipe of A-GNSS data, but this does not affect the outcome of most of the experiments, as we will see. Instead, for those experiments that require the presence of this feature, Apple devices are excluded.

7.3.2 Location Spoofing

Experiment 1 The first experiment involves spoofing the device position to a far away location in order to verify if any boundaries existed, out of which the devices coordinate would raise a flag or deny a fix. As mentioned in the previous section, this kind of attacks can be easily detected by applying the proposed security checks with information coming from other networks. For the sake of testing the implementation of these mechanisms, we kept the WiFi and data connections active on all smartphones.

We made sure all devices had previously obtained a fix at the coordinates Lat. 45.4056739, long. 11.898747 that is our lab in Padova, Italy. All aiding connections (WiFi, cellular, Bluetooth) were kept active, and the simulator user position was set to the coordinates Lat. 40.7484404, Long. -73.9878441 that is the Empire State Building in New York, USA. As reported in Tab. 7.2, all devices obtained a fix within 3 minutes, reporting the exact position selected in the simulation interface. This experiment suggests that the Open Service (OS) does not report a dramatic and sudden change of position, nor does it take into account the information coming from other sources in its final PVT solution.

7.3.3 Navigation Data Spoofing

The second experiment consists in the tampering with the GPS navigation message in the simulator to get an idea of how often it is observed and checked by the devices, and how they react to missing or non-compliant fields.

This experiment refers to the proposal of validating the authenticity of the navigation message by cross-checking it with the parameters obtained through A-GNSS. In case the proposed security check is already implemented, attacks that use spoofing signals with maliciously modified data will be detected.

In light of this consideration three further experiments were devised:

Experiment 2 We made sure the previous computed position was Lat. 40.7484404, Long. -73.9878441 (the Empire State Building) for all devices, with a complete spoofing signal. The command for wiping and updating the aiding navigation data (including the navigation message) was prompted. A spoofing signal was transmitted according to the coordinates Lat. -35.280937, long. 149.130009, that is Canberra, Australia, but this time the navigation data on the signal was replaced with all ones, except for the telemetry word, the handover word and the time indicators (time-of-week (TOW) and week number). All connections were kept active, and we waited to see how devices behaved in the presence of a signal partially deprived of its data. After a few minutes all devices computed the fake PVT, as reported in Tab. 7.2.

Experiment 3 We then pushed this result even further by completely wiping the navigation message (i.e., transmitting only the PRN code, without any data modulation). The previously computed position was forced to Lat. 40.7484404, Long. -73.9878441 (the Empire State Building), while the spoofing signal was built for a relatively close location: Lat. 40.641312, Long -73.778172 (the New York airport). We prompted the wipe off command for deleting navigation data from GPS Test. This time a subset of the devices lost the positioning capabilities, while the others had no trouble obtaining the false position fix.

The experiment shows that, for the sake of providing a usable PVT, some smartphones do not even react to the complete absence of the navigation message, since they do not necessarily need any of its content ([116], chapter 4). Although the pursuit of the availability of localization services is valuable in these devices, this result is rather alarming as it shows that even huge anomalies in the signal do not raise any alarm. On the other hand, the failure of the other smartphones in computing the position is most likely to be related either to the positioning algorithm itself or to a different implementation of A-GNSS, rather than to a threat-aware security check. The results are reported in Tab. 7.2.

Experiment 4 We devised another experiment, checking to which extent we could push the inconsistencies between the signal and the authentic location, still going undetected. This time the spoofing signal was generated for a further position: Lat. -35.280937, Long. 149.130009 (Camberra, Australia). All data were deleted, as in the previous experiment. We found this time that most devices would lose the positioning capability completely. However, as reported in Tab. 7.2, a few smartphones still managed to fix in the fake position.

Validity range violation As explained in Sec. 7.2.3, it is best practice for a receiver to check that the parameters received in the navigation message lay in the validity range.

Eccentricity within range We tested the presence of this validity check in the smartphones positioning protocol by devising another experiment. In particular, we tampered with the orbit eccentricity, that is one of the mentioned parameters having a specific validity range. The satellite constellation settings were manipulated in the software signal simulator, by substituting the eccentricity value for all satellite orbits with a constant value. The signal was thus generated from this fake constellation, and the navigation message was matched to the signal, to report the modified eccentricity value. Since we noticed that the devices download the ephemeris data from other aiding channels, when available, we switched them into airplane mode and prompted the cancellation and update of all navigation data. At first, we set a valid eccentricity value (ecc $= 0.02$) in order to verify which devices qualify for the experiment (i.e., those that respond to the wipe-and-update command for the navigation data). Forced to re-download the ephemeris from the only available source, that is the SIS, most smartphones obtained a position fix soon after, with the exception of those devices where the wipe command did not work as it should (Tab. 7.3 reports the details).

Eccentricity out of range The same procedure was repeated on those smartphones that obtained the PVT solution, but this time with an out-of-range orbit value: ecc $= 0.04$. It is interesting to notice how some of the devices behaved as before, while others did not calculate the position. This result suggests that a subset of the devices either do not perform a check on the valid range defined in the ICD, or they do not notify the application layer about the unexpected out-of-range parameter.

7.3.4 Time Spoofing

After successfully testing the smartphones against location spoofing, we investigated the effects of time incoherence between the GPS signal and the receivers clock. Our proposal pointed out how the time synchronization obtained through other connections (e.g., serving cellular network) is useful in order to discriminate between authentic and spoofed GNSS signals. The next experiment will test whether any security check of this kind is effectively used to defend the timeliness of GNSS positioning solutions, preventing the devices from accepting delayed or anticipated signals.

The reception time of the signal was thus changed, adding or subtracting a bias. At first we cautiously changed the time of just a few minutes and after obtaining positive results, we tested a few hours bias. We generated the GPS constellation for five hours in advance with respect to the current time, and we chose Miami (Lat. 25.7823907, Long. -80.2994992) as the receiver position (this corresponds to a combined time and position spoofing attack). We wiped the aiding data as usual and we observed the results reported in Tab. 7.4. The same sequence of operations was performed for five hours later than the current time.

The results seem to suggest that even a dramatic incoherence in the signal time does not influence the smartphones positioning capabilities, nor it raises a flag for a potential security threat. It is worth noting that Android and iOS devices behave differently when exposed to this kind of attack. While the former silently observe an incoherent GPS time, still delivering the fake position to the users, the latter uses the GPS time to correct their own clock, by default. Both approaches are dangerous from a security point of view. Android devices seem to never adjust their time according to the GNSS signal, but instead of recognizing it as ill-formed or malicious and notifying the user, they transparently

Table 7.2: Navigation data spoofing: Experiment results.

	Time to fix			
Smartphone	***Exp. 1***	***Exp. 2***	***Exp. 3***	***Exp. 4***
Apple iPhone 5	< 30 s	< 60 s	no fix	no fix
Apple iPhone 6s	< 120 s	< 30 s	no fix	no fix
Apple iPhone SE	< 60 s	< 60 s	no fix	no fix
Asus Nexus 7	< 120 s	< 30 s	no fix	no fix
Asus Zefone 2	< 30 s	< 180 s	< 30 s	no fix
Google Pixel	< 30 s	< 60 s	< 30 s	no fix
HTC one M9	< 60 s	no fix	no fix	no fix
Huawei Honor 8	< 30 s	< 30 s	< 30 s	< 30 s
Huawei Honor 9	< 30 s	< 30 s	< 120 s	< 120 s
Huawei p8 lite	< 60 s	< 180 s	no fix	no fix
Huawei p10 lite	< 30 s	< 120 s	no fix	no fix
LG Nexus 5	< 30 s	< 30 s	no fix	no fix
LG Nexus 5x	< 30 s	< 60 s	no fix	no fix
LG G6	< 30 s	< 60 s	< 60 s	no fix
LG G3	< 120 s	< 120 s	no fix	no fix
Motorola Moto G	< 30 s	< 30 s	no fix	no fix
OnePlus 2	< 60 s	< 120 s	no fix	no fix
OnePlus 5	< 60 s	< 120 s	no fix	no fix
Samsung S6 Edge	< 30 s	< 30 s	< 30 s	< 30 s
Samsung S6	< 30 s	< 60 s	< 60 s	< 240 s
Samsung S7 Edge	< 60 s	< 120 s	< 60 s	< 30 s
Xiaomi Mi 4c	< 60 s	< 120 s	no fix	no fix
Xiaomi Mi5	< 30 s	< 30 s	no fix	no fix
Xiaomi Mi6	< 120 s	no fix	no fix	no fix
Xiaomi Redmi Note 4x	< 60 s	< 120 s	< 30 s	no fix

Table 7.3: Validity range violation: spoofing signal with arbitrary eccentricity value. At first ecc = 0.02 (valid), then ecc = 0.04 (not valid)

	Spoofing success	
Smartphone	**within valid range**	**out of valid range**
Asus Nexus 7	yes	yes
Asus Zefone 2	n.a.	n.a.
Google Pixel	yes	no
HTC one M9	yes	yes
Huawei Honor 8	yes	no
Huawei Honor 9	yes	no
Huawei p8 lite	yes	yes
Huawei p10 lite	yes	yes
LG Nexus 5	yes	yes
LG Nexus 5x	yes	yes
LG G6	yes	yes
LG G3	yes	yes
Motorola Moto G	yes	yes
OnePlus 2	yes	yes
OnePlus 5	n.a.	n.a.
Samsung S6	yes	no
Samsung S7 Edge	yes	no
Xiaomi Mi 4c	yes	yes
Xiaomi Mi5	yes	yes
Xiaomi Mi6	yes	yes
Xiaomi Redmi Note 4x	yes	yes

accept the computed position. On the other hand, all iOS devices accepted the fake time, when a simple security check with the available networks would be enough to immediately point out a huge incoherence. This exposes the device to several annoying side effects e.g., the alarm will turn on at the wrong time and the calendar would send untimely notifications. Moreover, we noticed even more serious effects that caused the malfunctioning of the devices: the network and data connections were sometimes lost and the only way to retrieve connectivity was exposing them to the legitimate GPS signal.

Table 7.4: Time spoofing: experiment result

	Spoofing success	
Smartphone	***+ 5 h***	***- 5 h***
Apple iPhone 5	yes	yes
Apple iPhone 6s	yes	yes
Asus Nexus 7	yes	yes
Asus Zefone 2	no	no
Google Pixel	yes	yes
HTC one M9	no	yes
Huawei Honor 8	yes	yes
Huawei Honor 9	yes	yes
Huawei p8 lite	no	no
Huawei p10 lite	yes	yes
LG Nexus 5	yes	yes
LG Nexus 5x	yes	yes
LG G6	yes	yes
LG G4	yes	yes
Motorola Moto G	yes	yes
OnePlus OnePlus 2	yes	yes
OnePlus 5	yes	yes
Samsung S6	yes	no
Samsung S7 Edge	no	no
Xiaomi Mi 4c	no	yes
Xiaomi Mi5	no	yes
Xiaomi Mi6	yes	yes
Xiaomi Redmi Note 4x	yes	yes

7.3.5 Trajectory Spoofing

In another experiment we took into account dynamic rather than static spoofing. This is done to evaluate the presence of effective security checks based on sensor measurements, as proposed in Sec. 7.2.

We designed a path around our University and we used it in our simulator to generate the signal for a moving position, with a constant velocity of 20 km/h. The trajectory is represented in Fig. 7.3.

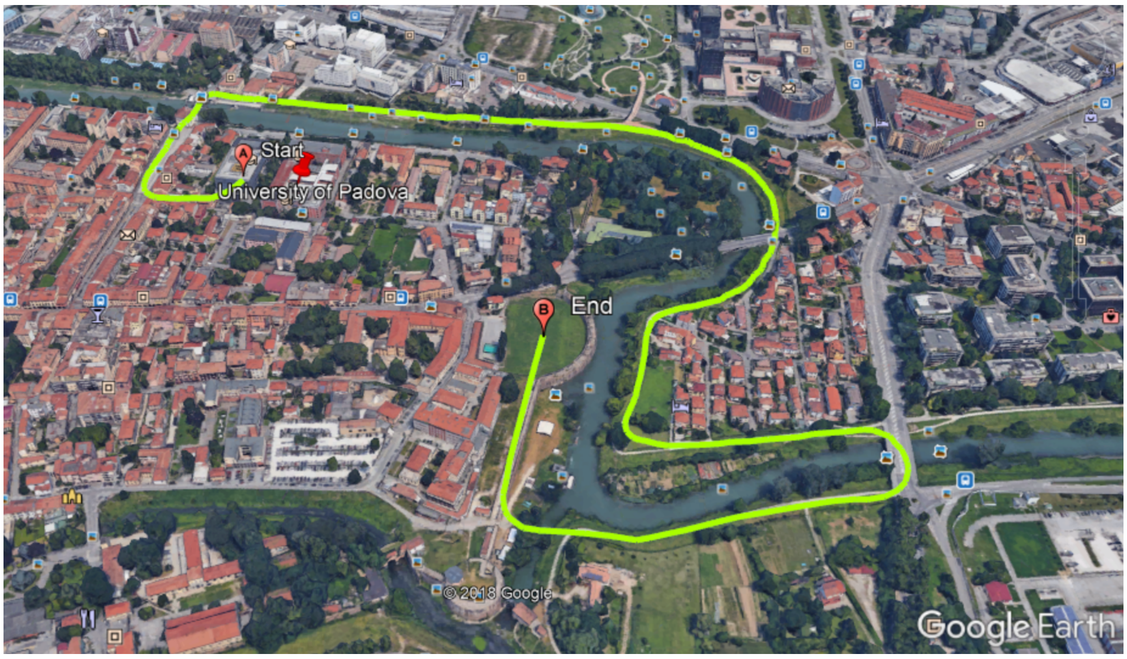


Figure 7.3: Spoofing trajectory

After wiping and updating the navigation data, we exposed the devices to the spoofing signal. All of them followed the spoofed trajectory at the expected velocity. It is interesting to notice that the devices were kept still on the table, where they were subject to no acceleration other than gravity. It is true that distinguishing between two different motion trajectories would be a complex problem, that requires to thoroughly measure the sensor data to derive adequate spoofing metrics. However, telling apart a moving trajectory from a static one is more straightforward, and it could be done by simply applying a threshold to the first and second order statistics of the sensor measurements in a certain time interval. This simple security check would at least raise a flag in case of massive incoherence between the position computed by GNSS and the motion captured by the sensors.

7.3.6 Testing the acceleration measurements

With this test we want to evaluate whether the accelerometer measurements can be exploited as a rough anti-spoofing indicator. An hour worth of measurements were collected from the same device (Huawei Honor 8), in three very different conditions: first the phone was left still on a table inside a room; then it was anchored to the dashboard of a car traveling at around 60 km/h in fluid traffic. For the third measurement the phone was kept inside the pocket while riding a bike.

Fig. 7.4 shows the cumulative density function (CDF) of the module of the linear acceleration vector (i.e., the acceleration referred to the Earth (x, y, z) coordinate system). The CDFs are just a mathematical tool that allows us to assess whether the different motion scenarios are statistically distinguishable. The figure shows that there is a visible differences between the phone standing still and the phone moving, although the two different motion scenarios (car and bike) can not be distinguished as easily.

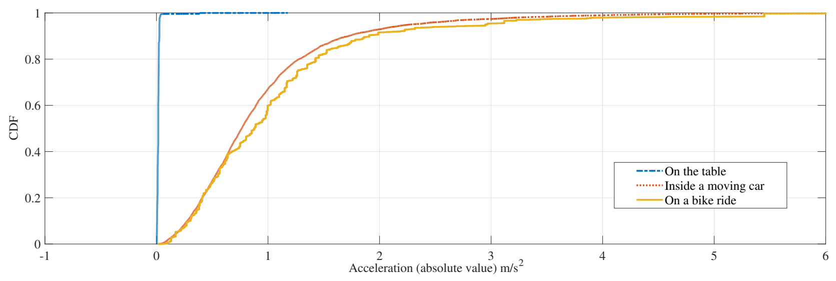


Figure 7.4: Cumulative density function of the module of the acceleration vector

This example shows that macro incoherence can be detected and reported to the application layer by a rough cross check with the device sensors.

7.3.7 Benchmark: a professional receiver

We performed all the experiments on a ublox receiver in order to understand the differences between professional equipment and mass market modules from the security point of view.

The ublox M8T allows to inject the time and an initial reference position from an external source. By reproducing experiment 1 we verified that the professional receiver is more sensitive to time synchronization: in case the *precise* external time reference mode was selected, spoofing the receiver with a loosely synchronized signal was not possible. However, when the *coarse* reference mode was used, the attack was successful.

The experiment 2, that involved the cancellation of the navigation data, leaving only telemetry word, handover word, and time references, was immediately successful. On the other hand, when the navigation message was completely removed, in experiment 3, the receiver did not compute the position fix. The same effect is obtained when the CRC fields of the navigation message are replaced with wrong values. We also tested experiment 4, verifying that no PVT was obtained when the spoofing signal was created for an out-of-range eccentricity value (ecc $= 0.04$).

The experiments showed that the professional GNSS receiver, differently from mobile devices, do not pursue availability at any cost, resulting in an enhanced resiliency to trivial spoofing attacks. This validates our claim that smartphones should take a step forward in positioning security, following at least some simple best practices that would improve spoofing resilience without great impacts on energy efficiency.

7.3.8 An Application Example: *Trusted Places*

Among the many applications that are based on GNSS measurements some are of particular interest from the security perspective: Google *Trusted Places*. It allows the user to unlock the phone without entering PIN, password or using fingerprints, if it is at a location that has previously been selected as a *trusted place*. In [127] it is reported that this feature is not secure against manipulation of GNSS signals. Indeed, we experimented the robustness of *trusted places* using a location spoofing attack and we were able to bypass the Android *Lock Screen*, provided that the false-position fix was previously obtained *with the screen unlocked*.

We believe that this application could sensibly benefit from the proposals of Sec. 7.2 as it might avoid the stealing of sensitive information from a device under spoofing attack.

7.3.9 Side effects of the experimental campaign

During the experimental campaign we noticed that the devices behavior (e.g., the time to fix) depended to some extent on the previous events (previous position fixes). Therefore it is important to remind the reader that even after validating the results with several observations, it is impossible to draw detailed conclusions, because no information is available on the inner functioning of the GNSS module and its interactions with the OS. However, beside the results reported in the previous paragraphs, we noticed some unexpected events and side effects in some of the tested smartphones, that seemed to be correlated with the experiment sessions, such as:

- sudden freeze of the OS, that required restarting the phone;
- persistence of the latest (fake) computed position for several hours after the end of the experiment and with the device being again exposed to the authentic signal;
- limited to iOS devices, persistence of the wrong time due to the exposure to a time spoofing attack;
- limited to iOS device, missing connection to the cellular network after exposure to a time/location spoofing attack;
- location-based notifications from several applications (e.g., weather, museum ticket suggestions, food tips, etc.) triggered by the induced false position.

7.4 Conclusions

This chapter has tackled the possible improvement of security policies in smartphone positioning. Some immediately available sources of side information were identified that can provide positioning redundancy and enhance the devices resilience to GNSS spoofing attacks. This proposal is based on different mechanisms (network connections, software checks and sensor measurements) that are available to all mobile devices and do not require major modification to the positioning algorithms and policies. We carried out an experimental campaign on several devices in order to identify whether the proposed ideas are already part of the state of the art positioning protocols. The results have shown that the suggested side information did not contribute to spoofing detection capabilities, as the application layer was never alerted of any anomaly when the phones were exposed to trivial spoofing attacks. We thus conclude that the security of mobile devices would greatly benefit from the implementation of the suggested operations. The proposed integrity checks would contribute to the protection of location based services, allowing the end user to safely exploit the emerging applications, that are currently exposed to the spoofing threat.

Chapter 8

Exploiting IMU measurements for anti-spoofing

8.1 Inertial Measurement Systems

As reported in [128], IMUs are in general composed by a combination of multiple accelerometers and gyroscopes that measure the force acting on it and it's angular velocity, respectively; some IMUs also integrate a magnetometer that measures the Earth's magnetic field. Spoofing detection techniques based on IMU measurements have been vastly used in safety critical applications such as aviation and maritime navigation, where GNSS spoofing is a serious threat. Several works have investigated the increase in security that high precision sensors can provide, [129, 130]. Since raw GNSS measurements are now available to Android smartphones applications [131] several works are focusing on GNSS integrated navigation with IMU in mobile devices, not only for performance improvement, but also for security enhancement. In [132, 133] a comparison of the performance improvement in the PNT between loosely and tightly coupled GNSS/IMU is carried out and compared to a system solely relying on GNSS. The result highlights how the tightly coupled implementation brings the better performance enhancement. Moreover, some relevant implementation issues are pointed out, such as the time synchronization between inertial sensor measurements and GNSS chipset data, data latency, and data sampling frequency. The authors of [134] perform quality assessment of measurements taken with IMU and GNSS chipsets in different mobile phone models. They point out how even measurements from low-cost IMUs of mobiles provide useful data for navigation integration.

The performance of low-cost accelerometers for anti-spoofing in aviation is reviewed in [135], where high frequency acceleration components are identified as a suitable source of randomness for authentication purposes, similarly to [129]. The work targets an attacker with imperfect information on the precise aircraft acceleration and develops a spoofing detection algorithm based on the decoupling of IMU and positioning obtained by GNSS, that allows a direct comparison of the acceleration for an unlimited time window. The proposed detection algorithm is reviewed in different application scenarios such as railways and automotive, wherein is found to be less effective due to the lower intensity of high frequency components in the acceleration process. As [134], also [135] shows that low-cost IMUs of mobile devices to provide useful corroborative information for anti-spoofing purposes. Spoofing detection in vehicular applications is investigated in [112], where a mobile device is used for comparing the absolute value of linear and angular acceleration with those obtained from the GNSS solution. This approach avoids the calibration of IMU and is invariant to manipulations to the device's initial orientation. The automotive scenario is also the target application of [136], wherein the proposed solution integrates data from GNSS, IMU, and odometer. Differently from [135] and [112], the comparison domain is position and not acceleration, and the detection statistics are obtained as the norm of the difference between position vectors (from GNSS and from IMU/odometer). The novelty in this approach is the idea of performing GNSS based sensor calibration at fixed time intervals only if the spoofing detection algorithm confirms that the GNSS solution is authentic. Several works in the literature have investigated the problem of anti-spoofing via integration and comparison with IMU data. The abundance of recent works on this idea confirms the interest of the community towards this topic. However, most of the reviewed papers focus on a specific application scenario with its restrictions and assumptions. Moreover, the design drivers that lead to the final spoofing detection algorithms and metrics sometimes lack generality. Some of these choices are, as an example: the comparison domain (position, velocity or acceleration); the trajectory assumptions and statistics; the setup of the IMU; the window of observation; the use of open or close loop in the Kalman filter, or even an intermediate approach; the detection metric.

8.2 System Model

The general problem addressed in this chapter regards GNSS anti-spoofing through the integration of multiple sources of positioning information. One of the aims of this investigation is to maintain sufficient generality so that the analysis can be extended to different scenarios and integrated with additional positioning sources.

From the previous chapters it is known that this source of PVT information is subject to spoofing through forgery of fake signals. On the contrary, tampering with the IMU data so that they are not representative of the legitimate movement is typically harder. An exception may be the magnetometer sensor, which can be tampered with by placing metal objects or magnets nearby the sensor, thus modifying the magnetic field. For the time being the focus will be solely on accelerometers and gyroscopes as IMU with the aim of deriving an optimal spoofing detection algorithm from the comparison of GNSS and inertial data.

Let us briefly review the of the considered sources of position information in the following.

8.2.1 Inertial Measurement Units

The general problem of navigation through sensor fusion, i.e., the integration of multiple positioning sources to obtain a more accurate and robust PVT solution, deals with data transformation between different coordinate frames. Following the approach of [137]:

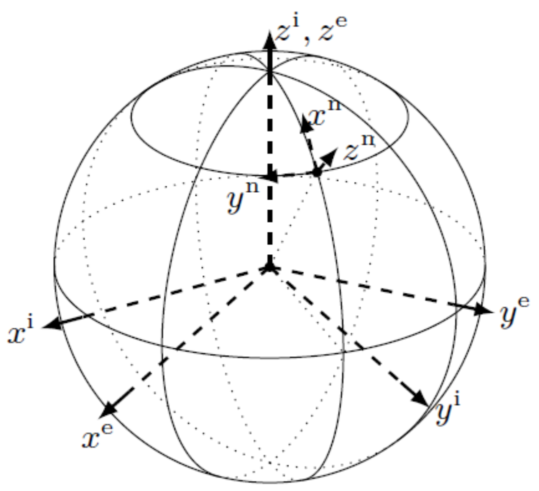


Figure 8.1: Earth, inertial and navigation frames, from [137].

Body frame, b: is the reference frame for IMU outputs. Its origin is located in the center of the accelerometer triad and the axes are generally aligned to the IMU case.

Navigation frame, n: is the target local geographical frame where we want to measure the object's PVT. In order to integrate the inertial IMU measurements we need to know the position and orientation of the b-frame with respect to the n-frame.

Inertial frame, i: is stationary with respect to the Earth. It has the origin in its center and the axis aligned with the stars;

Earth frame, e: it rotates with the Earth (origin in the Earth center and axis fixed with respect to the Earth).

As reported in [137] the IMU outputs acceleration and angular velocity of the body frame relative to the inertial frame, with measurements that are expressed in the body frame (i.e., with the body frame as reference basis).

Gyroscope measurements

The gyroscope measures the angular velocity of the body frame relative to the inertial frame, $\boldsymbol{\omega}_{\mathrm{ib}}^{(\mathrm{b})}$, where the superscript (b) indicates that the vector is expressed through coordinates in the b-frame. However for navigation purposes we are interested in $\boldsymbol{\omega}_{\mathrm{nb}}^{(\mathrm{b})}$ that is the angular velocity of the sensor relative to the navigation frame expressed in the b-frame, i.e.,

$$\boldsymbol{\omega}_{\mathrm{nb}}^{(\mathrm{b})}(t) = \boldsymbol{\omega}_{\mathrm{ib}}^{(\mathrm{b})}(t) - R^{(\mathrm{bn})}(t)(\boldsymbol{\omega}_{\mathrm{en}}^{(\mathrm{n})}(t) + \boldsymbol{\omega}_{\mathrm{ie}}^{(\mathrm{n})}), \tag{8.1}$$

where $R^{(\mathrm{bn})}$ is the rotation matrix from the n-frame to the b-frame, $\boldsymbol{\omega}_{(\mathrm{ie})}^{(\mathrm{n})}$ rad/s is the angular velocity of the earth frame relative to the inertial frame, and $\boldsymbol{\omega}_{\mathrm{en}}^{(\mathrm{n})}$ is the angular velocity of the n-frame relative to the e-frame.

In this work we assume that the n-frame is stationary with respect to the earth, thus $\boldsymbol{\omega}_{\mathrm{en}}^{(\mathrm{n})} = 0$. Moreover, since $|\boldsymbol{\omega}_{\mathrm{ie}}^{(\mathrm{n})}| \approx 7.29 \cdot 10^{-5}$ rad/s, we assume its contribution negligible.

For ease of notation, from (8.1), we define the time varying vector $\boldsymbol{\omega}(t)$ as

$$\boldsymbol{\omega}(t) \triangleq \boldsymbol{\omega}_{\mathrm{ib}}^{(\mathrm{b})} \approx \boldsymbol{\omega}_{\mathrm{nb}}^{(\mathrm{b})} \tag{8.2}$$

Accelerometer measurements

The accelerometer measures the force that acts on the sensor and computes the *specific* force, that is

$$\boldsymbol{a}_{\mathrm{i}}^{(\mathrm{b})}(t) = R^{(\mathrm{bn})}(t)(\boldsymbol{a}_{\mathrm{i}}^{(\mathrm{n})}(t) - \boldsymbol{g}^{(\mathrm{n})}), \tag{8.3}$$

where $\boldsymbol{g}^{(\mathrm{n})}$ is the gravitational acceleration and $\boldsymbol{a}_{\mathrm{i}}^{(\mathrm{n})}(t)$ is the acceleration of the device relative to the i-frame. For navigation purposes we are interested in $\boldsymbol{a}_{\mathrm{n}}^{(\mathrm{n})}$, the acceleration of the device relative to the n-frame. The relationship between $\boldsymbol{a}_{\mathrm{n}}^{(\mathrm{n})}$ and $\boldsymbol{a}_{\mathrm{i}}^{(\mathrm{n})}$ is [137]

$$\boldsymbol{a}_{\mathrm{i}}^{(\mathrm{n})}(t) = \boldsymbol{a}_{\mathrm{n}}^{(\mathrm{n})}(t) + 2\boldsymbol{\omega}_{\mathrm{ei}}^{(\mathrm{n})} \times \boldsymbol{v}(t) + \boldsymbol{\omega}_{\mathrm{ei}}^{(\mathrm{n})} \times \boldsymbol{\omega}_{\mathrm{ei}}^{(\mathrm{n})} \times \boldsymbol{p}(t), \tag{8.4}$$

where $\boldsymbol{p}$ and $\boldsymbol{v}$ are position and velocity of the device relative to the navigation frame.

In (8.4) the angular velocity of the Earth has been assumed constant and the navigation frame is fixed to the earth frame, which is reasonable when the travelled distance is negligible with respect to the earth radius. This formulation is derived by using the relation between rotating coordinate frames. The last term of the sum represents the centrifugal acceleration, while the second is the Coriolis acceleration. The former is typically absorbed in the gravity vector and has magnitude of around $3.39 \cdot 10^{-2}\mathrm{m}/s^2$ while the latter depends on the velocity of the object on which the IMU is mounted, and has magnitude in the order of 10^{-3} for a speed of 120 km/h.

In this work, we regard these last two contributions negligible and, therefore, for ease of notation we define the time varying vector $\boldsymbol{a}(t)$

$$\boldsymbol{a}(t) \triangleq \boldsymbol{a}_{\mathrm{n}}^{(\mathrm{n})}(t) \approx \boldsymbol{a}_{\mathrm{i}}^{(\mathrm{n})}(t). \tag{8.5}$$

Moreover, the physical relationships between $\boldsymbol{a}$, $\boldsymbol{p}$ and $\boldsymbol{v}$ are

$$\boldsymbol{v}(t) = \frac{\partial \boldsymbol{p}(t)}{\partial t}, \quad \boldsymbol{a}(t) = \frac{\partial \boldsymbol{v}(t)}{\partial t}. \tag{8.6}$$

8.2.2 Constraint on Acceleration and Position

By integrating (8.6), we get

$$\boldsymbol{p}(t_0 + \tau) = \int_{t_0}^{t_0+\tau} \int_{t_0}^{z} \boldsymbol{a}(w)\, dw\, dz + \tau \boldsymbol{v}_{t_0} + \boldsymbol{p}_{t_0}, \quad \tau > 0 \tag{8.7}$$

where $\boldsymbol{v}_{t_0}$ and $\boldsymbol{p}_{t_0}$ are the initial conditions. Let B be the bandwidth of $a(t)$. If we are given samples $a(nT_a)$, $n \in \mathbb{Z}$, with $T_a > 2B$, i.e., that satisfy the sampling theorem, then we can write

$$\boldsymbol{p}(t_0 + \tau) = \int_{t_0}^{t_0+\tau} \int_{t_0}^{z} \sum_{k=-\infty}^{+\infty} \boldsymbol{a}(kT_a + t_0)\, \mathrm{sinc}\left(F_a(w - t_0) - k\right)\, dw\, dz + \boldsymbol{c}(\tau), \tag{8.8}$$

with $F_a = 1/T_a$ and

$$\boldsymbol{c}(\tau) \triangleq \tau \boldsymbol{v}_{t_0} + \boldsymbol{p}_{t_0}$$

. By exchanging integrals with summation we get

$$\boldsymbol{p}(t_0 + \tau) = \sum_{k=-\infty}^{+\infty} \boldsymbol{a}(kT_a + t_0) \int_{t_0}^{t_0+t} \int_{t_0}^{z} \operatorname{sinc}\left(F_a(w - t_0) - k\right)\, dw\, dz + \boldsymbol{c}(\tau). \tag{8.9}$$

By sampling $\tau = nT_p$ we get

$$\boldsymbol{p}(t_0 + nT_p) = \sum_{k=-\infty}^{+\infty} \boldsymbol{a}(kT_a) s_n(k) + \boldsymbol{c}(nT_p) \tag{8.10}$$

$$s_n(k) \triangleq \int_{t_0}^{t_0+nT_p} \int_{t_0}^{z} \operatorname{sinc}\left(F_a(w - t_0) - k\right)\, dw\, dz. \tag{8.11}$$

By changing integration variable in both integrals we get

$$\begin{aligned} s_n(k) &= \frac{1}{F_a} \int_{t_0}^{t_0+nT_p} \left[\operatorname{Si}(F_a(z - t_0) - k) + \operatorname{Si}(k)\right]\, dz, \\ &= \frac{1}{(\pi F_a)^2} \left[\cos(B) + B \operatorname{Si}(B) - \cos(A) - A \operatorname{Si}(A)\right] - \frac{1}{\pi F_a} \tau \operatorname{Si}(A), \end{aligned} \tag{8.12}$$

where

$$A = \pi(F_a t_0 - k) \tag{8.13}$$

$$B = \pi(F_a t_0 + F_a n T_p - k), \tag{8.14}$$

and $\operatorname{Si}(\cdot)$ is the sine integral

$$\operatorname{Si}(x) \triangleq \int_0^x \operatorname{sinc}(\eta)\, d\eta. \tag{8.15}$$

Using N total samples, $\boldsymbol{a}(kT_a)$, $k = 0, \ldots, N-1$, $N > N_a$, we can write a relation for $N - N_a + 1$ samples $\boldsymbol{p}(nT_p)$. Let $\boldsymbol{a}$, $\boldsymbol{p}$ and $\boldsymbol{c}$ be vectors collecting all considered time samples

$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{a}(T_a) \\ \boldsymbol{a}(2T_a) \\ \cdots \\ \boldsymbol{a}(NT_a) \end{bmatrix}, \quad \boldsymbol{p} = \begin{bmatrix} \boldsymbol{p}(T_p) \\ \boldsymbol{p}(2T_p) \\ \cdots \\ \boldsymbol{p}((N - N_a + 1)T_p) \end{bmatrix}, \tag{8.16}$$

$$\boldsymbol{c} = \begin{bmatrix} T_p \boldsymbol{v}_0 + p_0 \\ 2T_p \boldsymbol{v}(T_p) + \boldsymbol{p}(T_p) \\ \cdots \\ (N - N_a + 1)T_p \boldsymbol{v}((N - N_a)T_p) + \boldsymbol{p}((N - N_a)T_p) \end{bmatrix}. \tag{8.17}$$

We can then write

$$\mathbf{p} = A\mathbf{a} + \mathbf{c}, \tag{8.18}$$

where A is a $(N - N_a + 1) \times N_a$ matrix suitably filled with $s_n(k)$ entries.

8.2.3 Constraint on Orientation

The acceleration $a(t)$ of equation (8.7) is not among the measured quantities. Indeed the measured acceleration $a_b(t)$ is expressed with respect to the body frame rather than the navigation frame. In order to express $a(t)$ as a function of $a_b(t)$ the former must be rotated and the entity and direction of this rotation is given by the orientation of the navigation frame with respect to the body frame.

Orientation can be parametrized in different ways, differing in the number of parameters, the uniqueness and the singularities. Let us consider unit quaternions, one of the most widely used orientation parametrization in estimation problems, according to [137]. Unit quaternions are a 4-dimensional representation of orientation:

$$\mathbf{q} = (q_0, q_1, q_2, q_3)^T = \begin{pmatrix} q_0 \\ \mathbf{q_v} \end{pmatrix}, \ \mathbf{q_v} \in \mathbb{R}^3, \ ||\mathbf{q}||_2 = 1 \tag{8.19}$$

A rotation of a vector in $\mathbb{R}^3$ is a change of its direction while its length remains constant. The rotation of $\mathbf{x_a}$ into $\mathbf{x_b}$ can be expressed with unit quaternions as:

$$\mathbf{x_b} = \mathbf{q^{ba}} \odot \mathbf{x_a} \odot (\mathbf{q^{ba}})^c \tag{8.20}$$

where the $\odot$ represents quaternion multiplication, that can be expressed in matrix form (see [137] for the derivation).

Rotations in $\mathbb{R}^3$ form the special orthogonal group, $SO(3)$, that is a matrix *Lie group*. As reported in [137], this allows to represent an orientation deviation with an exponential map over rotation vectors. An orientation with respect to the navigation frame, $\mathbf{q_t^{nb}}$ is thus represented in terms of a linearization point ($\mathbf{\tilde{q}_t^{nb}}$) and an orientation deviation parametrized by a rotation vector, $\boldsymbol{\eta_t}$, expressed in the body frame:

$$\mathbf{q_t^{nb}} = \exp\left(\frac{\bar{\boldsymbol{\eta}}_{\boldsymbol{t}}}{2}\right) \odot \mathbf{\tilde{q}_t^{nb}} \tag{8.21}$$

where $\bar{\boldsymbol{\eta}}_{\boldsymbol{t}} = (0, \boldsymbol{\eta_t}^T)^T$, and $\boldsymbol{\eta_t} = \mathbf{n}\alpha$ is a rotation vector, parametrized by a unit vector, $\mathbf{n}$ and rotation Euler angles α. The exponential operation is defined by:

$$\exp(\bar{\boldsymbol{\eta}}) = \cos||\boldsymbol{\eta}||_2 + \frac{\bar{\boldsymbol{\eta}}}{||\bar{\boldsymbol{\eta}}||} \sin||\boldsymbol{\eta}||_2 \tag{8.22}$$

In our case we are interested in expressing the orientation in time as a function of the angular velocity and the initial orientation. Therefore equation (8.21) will be used, where the reference orientation $\mathbf{\tilde{q}_t^{nb}}$ is the initial orientation at time t_0, and the rotation vector is represented by the angle displacement, i.e., the time integral of the angular speed:

$$\mathbf{q^{nb}}(t) = \exp\left(\frac{\bar{\boldsymbol{\eta}}(t)}{2}\right) \odot \mathbf{\tilde{q}^{nb}}(t_0) \tag{8.23}$$

with

$$\boldsymbol{\eta}(t) = \int_{t_0}^{t} \boldsymbol{\omega}(t)dt \tag{8.24}$$

The acceleration in the navigation frame can thus be expressed in the following way with respect to the acceleration in the body frame and the relative orientation of the two coordinate frames:

$$\mathbf{a}^{\mathbf{n}}(t) = \mathbf{q}^{\mathbf{nb}}(t) \odot \mathbf{a}^{\mathbf{b}}(t) \odot (\mathbf{q}^{\mathbf{nb}}(t))^c. \tag{8.25}$$

8.3 Measurement Error Models

Data from gyroscopes and accelerometers are corrupted by measurement noise, [138]. By collecting data from a stationary IMU standing on a flat surface the gyroscope is expected to measure only the earth rotation, while the accelerometers should measure the resulting acceleration that accounts for gravity and the centrifugal force. Over some tens of seconds the data seem to fit well a Gaussian distribution with non-zero mean.

8.3.1 Gyroscope

In general the noise can be divided in two distinct contributions: a slowly time varying bias $\boldsymbol{\delta}_{\omega,t}$ and a white noise component $\boldsymbol{e}_{\omega,t} \sim \mathcal{N}(0, \Sigma_\omega)$, with Σ_ω a 3×3 diagonal matrix. The subscript t denotes discrete time samples. Therefore the measures can be written as

$$\boldsymbol{y}_{\omega,t} = \boldsymbol{\omega}(nT_\omega) + \delta_{\omega,t} + e_{\omega,t}, \tag{8.26}$$

where T_ω is the sampling period of the gyroscope.

There are in general two approaches to model the bias: either consider it constant in the time interval of the measurements or consider it slowly time varying and model it as a random walk

$$\boldsymbol{\delta}_{\omega,t+1} = \boldsymbol{\delta}_{\omega,t} + \boldsymbol{e}_{\delta_{\omega,t},t}, \tag{8.27}$$

with $\boldsymbol{e}_{\delta_{\omega,t},t} \sim \mathcal{N}\left(0, \Sigma_{\delta_{\omega,t}}\right)$ and $\Sigma_{\delta_{\omega,t}}$ 3×3 diagonal.

This model fits well the experimental data and can be verified by the means of the Allan variance.

8.3.2 Accelerometer

For the accelerometer the same considerations hold, and the noise has two contributions: a bias that is slowly time varying and a white noise. The measurement model for the accelerometer is:

$$\boldsymbol{y}_{a,t} = \boldsymbol{a}_{\mathrm{i}}^{(\mathrm{b})}(t) + \boldsymbol{\delta}_{a,t} + \boldsymbol{e}_{a,t}, \tag{8.28}$$

where T_a is the sampling period of the accelerometer, and $\boldsymbol{e}_{a,t} \sim \mathcal{N}\left(0, \Sigma_{\delta_{a,t}}\right)$, with again $\Sigma_{\delta_{a,t}}$ a 3×3 diagonal matrix.

8.3.3 Global Navigation Satellite System

The lower level output of the GNSS module are the pseudorange measurements $\rho_t^{(s)}$ and carrier phase measurements $\varphi_t^{(s)}$, where the superscript s denotes the number of satellites in view. From the raw measurements and the ephemeris data we can compute PVT.

GNSS measurements are corrupted by additive noise, i.e.,

$$\boldsymbol{y}_{p,t} = \boldsymbol{p}(nT_p) + \boldsymbol{e}_{p,t}, \tag{8.29}$$

where T_p is the sampling period of the GNSS module. A reasonable model for the additive error process is a Gauss-Markov process defined as follows

$$\boldsymbol{e}_{p,t+1} = \exp(-\beta T_p)\boldsymbol{e}_{p,t} + \nu_t, \tag{8.30}$$

where β is a parameter describing the correlation between successive samples, $\nu_t \sim \mathcal{N}(0, \Sigma_p)$. Parameters β, T_p and Σ_p are tabulated in [139] [140].

8.4 Hypothesis Testing

The aim of this analysis is to find a mechanism that allows to state whether the GNSS module is under spoofing or not. The optimal solution requires the knowledge of the joint statistic of the raw measurements from both the IMU and GNSS module, under the two hypotheses

$\mathcal{H}_0$: GNSS module is not under spoofing

$\mathcal{H}_1$: GNSS module is under spoofing,

where with *spoofing* we indicate any attack inducing the victim to compute a PVT different from the actual one. Let $\boldsymbol{y}$ the vector collecting all N measures from sensors as

$$\boldsymbol{y} \triangleq \begin{bmatrix} \boldsymbol{y}_{\omega,0}^T & \dots & \boldsymbol{y}_{\omega,N}^T & \boldsymbol{y}_{a,0}^T & \dots & \boldsymbol{y}_{a,N}^T & \boldsymbol{y}_{p,0}^T & \dots & \boldsymbol{y}_{p,N}^T \end{bmatrix}^T. \tag{8.31}$$

Let then $\boldsymbol{\vartheta}$ be the vector collecting all the unknown quantities

$$\boldsymbol{\vartheta} \triangleq \begin{bmatrix} \boldsymbol{a}^T & \boldsymbol{p}^T & \boldsymbol{q}_1^T & \dots & \boldsymbol{q}_N^T & \boldsymbol{d} \end{bmatrix}, \tag{8.32}$$

where $\boldsymbol{d}$ collects all unknown parameters of the measures' Probability Density Function (PDF).

We then denote the likelihood of the two hypotheses as

$$\mathcal{L}(\boldsymbol{y}, \boldsymbol{\vartheta}) = \frac{p(\mathbf{y}, \boldsymbol{\vartheta}|\mathcal{H}_0)}{p(\mathbf{y}, \boldsymbol{\vartheta}|\mathcal{H}_1)}, \tag{8.33}$$

Then, following the Neyman-Pearson theory, we take the optimal decision by thresholding $\mathcal{L}$, i.e.,

$$\hat{\mathcal{H}} = \begin{cases} \mathcal{H}_0 & \text{if} \quad \mathcal{L}(\boldsymbol{y}, \boldsymbol{\vartheta}) > \gamma \\ \mathcal{H}_1 & \text{if} \quad \mathcal{L}(\boldsymbol{y}, \boldsymbol{\vartheta}) < \gamma \end{cases}, \tag{8.34}$$

where γ is the decision threshold.

Under $\mathcal{H}_1$ we consider IMU measurements to be independent of GNSS measurements, since the latter is under the attacker's control, while the former is not. We can write then

$$p(\mathbf{y}, \boldsymbol{\vartheta}|\mathcal{H}_1) = p(\mathbf{y}_\mathrm{I}, \boldsymbol{\vartheta}|\mathcal{H}_1)p(\mathbf{y}_\mathrm{G}, \boldsymbol{\vartheta}|\mathcal{H}_1), \tag{8.35}$$

where

$$\boldsymbol{y}_{\mathrm{I}} \triangleq \begin{bmatrix} \boldsymbol{y}_{\omega,0}^T & \cdots & \boldsymbol{y}_{\omega,N}^T & \boldsymbol{y}_{a,0}^T & \cdots & \boldsymbol{y}_{a,N}^T & \end{bmatrix}^T \tag{8.36}$$

$$\boldsymbol{y}_{\mathrm{G}} \triangleq \begin{bmatrix} \boldsymbol{y}_{p,0}^T & \cdots & \boldsymbol{y}_{p,N}^T \end{bmatrix}^T \tag{8.37}$$

Given the fact that we give the attacker complete freedom, we cannot assume any particular distribution $p(\boldsymbol{y}_G, \boldsymbol{\vartheta}|\mathcal{H}_1)$ and must therefore ignore it from 8.33. Then, $\mathcal{L}$ becomes

$$\mathcal{L}(\boldsymbol{y}, \boldsymbol{\vartheta}) = \frac{p(\mathbf{y}, \boldsymbol{\vartheta}|\mathcal{H}_0)}{p(\mathbf{y}_{\mathrm{I}}, \boldsymbol{\vartheta}|\mathcal{H}_1)} \underset{\mathcal{H}_1}{\overset{\mathcal{H}_0}{\gtrless}} \gamma. \tag{8.38}$$

However, we need some prior knowledge on the joint PDF of $\boldsymbol{\vartheta}$, as $\mathcal{L}$ depends on $p(\boldsymbol{\vartheta})$

$$\mathcal{L} = \frac{\int_{\boldsymbol{\vartheta}} p(\mathbf{y}|\boldsymbol{\vartheta}, \mathcal{H}_0) p(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta}}{\int_{\boldsymbol{\vartheta}} p(\mathbf{y}_{\mathrm{I}}|\boldsymbol{\vartheta}, \mathcal{H}_1) p(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta}}. \tag{8.39}$$

8.4.1 Generalized Likelihood Ratio Test

Following known results in detection theory [141], we substitute $\boldsymbol{\vartheta}$ in (8.38) with its maximum likelihood estimation under the two hypotheses, i.e.,

$$\hat{\boldsymbol{\vartheta}}_0 \triangleq \arg\max p(\mathbf{y}|\boldsymbol{\vartheta}, \mathcal{H}_0), \tag{8.40}$$

$$\hat{\boldsymbol{\vartheta}}_1 \triangleq \arg\max p(\mathbf{y}_{\mathrm{I}}|\boldsymbol{\vartheta}, \mathcal{H}_1). \tag{8.41}$$

We then compute

$$\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{\vartheta}}_0, \hat{\boldsymbol{\vartheta}}_1) = \frac{p(\mathbf{y}, \hat{\boldsymbol{\vartheta}}_0|\mathcal{H}_0)}{p(\mathbf{y}_{\mathrm{I}}, \hat{\boldsymbol{\vartheta}}_1|\mathcal{H}_1)} \underset{\mathcal{H}_1}{\overset{\mathcal{H}_0}{\gtrless}} \gamma, \tag{8.42}$$

or directly evaluate the likelihood

$$\frac{\max\limits_{\boldsymbol{\vartheta}} p(\mathbf{y}|\boldsymbol{\vartheta}, \mathcal{H}_0)}{\max\limits_{\boldsymbol{\vartheta}} p(\mathbf{y}_{\mathrm{I}}|\boldsymbol{\vartheta}, \mathcal{H}_1)} \underset{\mathcal{H}_1}{\overset{\mathcal{H}_0}{\gtrless}} \gamma. \tag{8.43}$$

We now explicitly write (8.40).

Let Σ_a, Σ_p and Σ_ω be the covariance matrices of $\boldsymbol{y}_a$, $\boldsymbol{y}_p$ and $\boldsymbol{y}_\omega$ respectively. We can write

$$\begin{aligned} &\arg\max_{\boldsymbol{\vartheta}} p(\mathbf{y}|\boldsymbol{\vartheta}, \mathcal{H}_0) = \\ &\arg\min_{\boldsymbol{\vartheta}} (\boldsymbol{y}_a - \mathbf{a})^t \Sigma_a (\boldsymbol{y}_a - \boldsymbol{a}) + (\boldsymbol{y}_p - \mathbf{p})^t \Sigma_p (\boldsymbol{y}_p - \mathbf{p}) + (\boldsymbol{y}_\omega - \boldsymbol{\omega})^t \Sigma_\omega (\boldsymbol{\omega} - \boldsymbol{\omega}) \end{aligned} \tag{8.44}$$

Subject to constraints (8.18) and (8.25).

Similarly, for (8.41) it is

$$\arg\max_{\boldsymbol{\vartheta}} p(\mathbf{y}_{\mathrm{I}}|\boldsymbol{\vartheta}, \mathcal{H}_1) = \arg\min_{\boldsymbol{\vartheta}} (\boldsymbol{y}_a - \mathbf{a})^t \Sigma_a (\boldsymbol{y}_a - \boldsymbol{a}) + (\boldsymbol{y}_\omega - \boldsymbol{\omega})^t \Sigma_\omega (\boldsymbol{\omega} - \boldsymbol{\omega}). \tag{8.45}$$

Subject to constraints (8.18) and (8.25).

8.4.2 Algorithms for GLRT

Solving (8.40) and (8.41) exactly is not algorithmically feasible since, for example, the physical models linking $\boldsymbol{\vartheta}$ are continuous-time and we are dealing instead with discrete-time samples. We then identified two sub-optimal approaches.

1. KF estimation,
2. Quadratic programming.

All these approaches rely on the KF, which we recall here briefly.

The KF defines a state $\boldsymbol{x}_k$ at iteration k, a measurement vector $\boldsymbol{z}_k$, a state-update function $f(\cdot)$ and a measurement function $h(\cdot)$ such that

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}_k) \tag{8.46}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k, \boldsymbol{v}_k), \tag{8.47}$$

where u_k is the control input, $\boldsymbol{w}_k$ is process noise, and $\boldsymbol{v}_k$ is measurement noise. In our case we don't need $\boldsymbol{u}_k$, therefore, in the following, we ignore it. The output of the KF algorithm is an estimation $\{\hat{\boldsymbol{x}}_k\}_k$ of the state.

A special case of KF arises when $f(\cdot)$ and/or $h(\cdot)$ are linear. In this case the KF model becomes

$$\boldsymbol{x}_{k+1} = F\boldsymbol{x}_k + \boldsymbol{w}_k \tag{8.48}$$

$$\boldsymbol{z}_k = H\boldsymbol{x}_k + \boldsymbol{w}_k. \tag{8.49}$$

We usually refer to the general case with Extended Kalman Filter (EKF).

Challenges in KF design are the state choice and parameter tuning, e.g. the statistical description of the process noise. Currently we have two types of KFs readily available.

1. **Input**: accelerometer and gyroscope measurements. **Output**: orientation and angular velocity estimations.
2. **Input**: accelerometer, gyroscope and GNSS measurements. **Output**: orientation, position, velocity, and acceleration estimations.

Kalman Filter Estimation

We can solve (8.40) and (8.41) using two distinct KFs. For $\mathcal{H}_0$ we need a KF which comprises in the states at least position, angular velocity and acceleration, while for $\mathcal{H}_1$ the state must comprise only acceleration and angular velocity, since position measurements are not considered in this case. With the state estimates we then compute (8.42).

The sub-optimality comes from the fact that all the physical relations in the state-update function are discretized.

This approach requires the design of two KFs that are not standard navigation and sensor fusion KFs. Details of this method will be provided in future reports.

Quadratic Programming

We observe that (8.44) and (8.45) are quadratic objective function and (8.25) is a linear constraint. Orientation constraints instead, are non linear. We then tackle non linearity by separately estimating orientation so that we are left with a quadratic programming problem, that can be solved efficiently.

Orientation Estimation Orientation estimation can be performed starting from accelerometer and gyroscope measurements. We use a state-of-the-art KF already present in the Matlab libraries (Type 1 KF in Section 8.4.2). The state and state-update function are

$$\begin{bmatrix} \varepsilon_k \\ \boldsymbol{\delta}_{\omega,k} \\ \boldsymbol{a}_k^{(\varepsilon)} \end{bmatrix} = F \begin{bmatrix} \varepsilon_{k-1} \\ \boldsymbol{\delta}_{\omega,k-1} \\ \boldsymbol{a}_{k-1}^{(\varepsilon)} \end{bmatrix} + \boldsymbol{w}_k, \tag{8.50}$$

where ε_k is the error between the estimated orientation and true orientation in degree, $\boldsymbol{\delta}_{\omega,k}$ is the gyroscope bias, and $\boldsymbol{a}_k^{(\varepsilon)}$ is the error between the estimated acceleration and the true acceleration. The measurement process is defined as

$$\boldsymbol{\xi}_k^g = H\boldsymbol{x}_k + \boldsymbol{w}_k, \tag{8.51}$$

where, $\boldsymbol{\xi}_k^g$ is the difference between the accelerometer and gyroscope estimates of the gravity vector. For the derivation of matrices H and F, we refer to [142].

State estimations are then processed [142, Section 5] in order to obtain an estimate of $\boldsymbol{q}^{\text{nb}}$, which we denote with $\hat{\boldsymbol{q}}^{\text{nb}}$. Then, since a rigid rotation does not change the statistics of $\boldsymbol{\delta}_{a,t}$ and $\boldsymbol{e}_{a,t}$ in (8.28), we obtain the new acceleration measurement model

$$\boldsymbol{y}_a^{\text{n}} = \boldsymbol{a}(t) + \boldsymbol{\delta}_{a,t} + e_{a,t}, \tag{8.52}$$

that is, we have rotated the measurements from the body frame to the navigation frame.

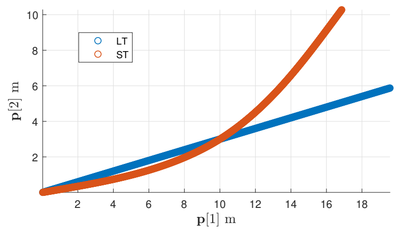
Quadratic Programming After orientation estimation we are left with the following quadratic optimization problem under hypothesis $\mathcal{H}_0$

$$\begin{aligned} [\hat{\boldsymbol{a}}_{\mathcal{H}_0}, \hat{\boldsymbol{p}}_{\mathcal{H}_0}] =& \arg\min_{\boldsymbol{a},\boldsymbol{p}} \, (\boldsymbol{y}_a^{\text{n}} - \mathbf{a})^t \Sigma_a (\boldsymbol{y}_a^{\text{n}} - \boldsymbol{a}) + (\boldsymbol{y}_p - \mathbf{p})^t \Sigma_p (\boldsymbol{y}_p - \mathbf{p}) \\ & \text{subject to} \quad \boldsymbol{p} = A\boldsymbol{a} + \boldsymbol{c}. \end{aligned} \tag{8.53}$$

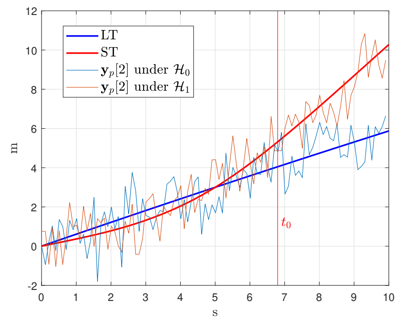
Similarly, under $\mathcal{H}_1$, we have

$$\begin{aligned} [\hat{\boldsymbol{a}}_{\mathcal{H}_1}, \hat{\boldsymbol{p}}_{\mathcal{H}_1}] =& \arg\min_{\boldsymbol{a},\boldsymbol{p}} \, (\boldsymbol{y}_a^{\text{n}} - \mathbf{a})^t \Sigma_a (\boldsymbol{y}_a^{\text{n}} - \boldsymbol{a}) \\ & \text{subject to} \quad \boldsymbol{p} = A\boldsymbol{a} + \boldsymbol{c}. \end{aligned} \tag{8.54}$$

Both optimizations under the two hypothesis can be solved using the `quadprog` function of Matlab. We then apply (8.42) and perform detection.



(a) Scatter plot of $\boldsymbol{p}[1]$ and $\boldsymbol{p}[2]$ of the spoofed and authentic trajectory. $\vartheta = \pi/6$.



(b) $\boldsymbol{p}[2]$ of the spoofed and authentic trajectory, versus time. Thinner lines are the corresponding measurements.

Figure 8.2: Simulation scenario.

8.4.3 Innovation Testing

Innovation testing is a spoofing detection approach that exploits the KF designed for sensor fusion and navigation. Typically, such KFs have position, velocity and orientation as state, and IMU and GNSS as measurements (type 2 KF in Section 8.4.2).

The innovation step, in any linear KF (a similar expression holds for EKF), is

$$\boldsymbol{i}_k = \boldsymbol{z}_k - H\hat{\boldsymbol{x}}_{k|k-1}, \tag{8.55}$$

where $\hat{\boldsymbol{x}}_{k|k-1}$ is a prediction of the current state. The covariance matrix of $\boldsymbol{i}$ is known [143] and denoted by P_k. Then, by normalizing $\boldsymbol{i}_k$ by its covariance matrix, we obtain the test statistic

$$\beta_k = \boldsymbol{i}_k^t P_k \boldsymbol{i}_k, \tag{8.56}$$

which can be shown to be Chi-squared distributed with as many degree of freedom as the dimension of the measurement vector $\boldsymbol{z}$ [143].

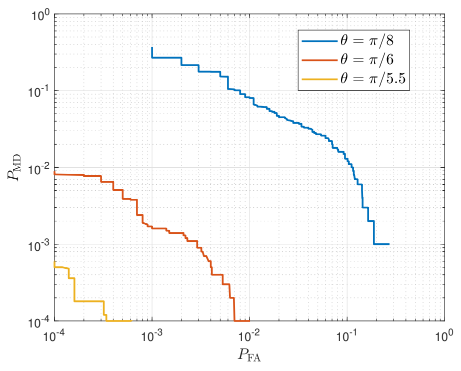
Anti-spoofing based on innovation testing exploits the fact that under spoofing, $\boldsymbol{x}_{k|\hat{k}-1}$ is no longer a good prediction of the state, since it is a prediction coming from IMU (not under spoofing) and GNSS, which is under spoofing. Then under spoofing β_k is no longer Chi-squared distributed and the spoofing detection is performed according to

$$\hat{\mathcal{H}} = \begin{cases} \mathcal{H}_0 & \beta_k \sim \chi^2 \\ \mathcal{H}_1 & \beta_k \not\sim \chi^2. \end{cases} \tag{8.57}$$

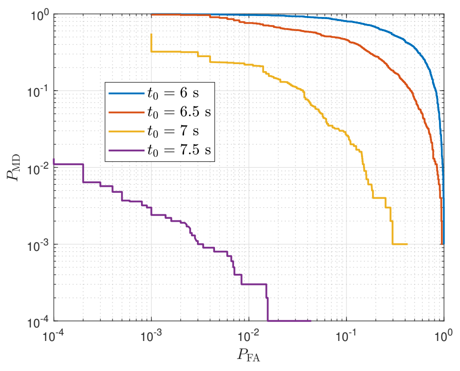
In [143] the statistic of β_k under a spoofing scenario is derived. Innovation testing is common in the literature, especially in aviation scenarios [130, 144].

8.5 Preliminary Results

We simulate a simple spoofing scenario by designing a LT, i.e., the trajectory that the user physically follows, and a ST, i.e., the trajectory that the spoofer induces to the user by performing a spoofing attack. We then generate the corresponding IMU and GNSS



(a) DET curves for different ϑ.



(b) DET curves for different t_0.

Figure 8.3: DET curves for spoofing detection, varying t_0 and ϑ

measurements and apply the detection algorithm of Section 8.4.2. Let us recall that under $\mathcal{H}_0$ both GNSS and IMU follow the LT, while under $\mathcal{H}_1$ GNSS follows the ST.

The ST starts from the same position as the LT, but then diverges by an angle ϑ.

Fig. 8.2 shows the LT and the ST. Fig. 8.2a shows a 2D scatter plot of $\boldsymbol{p}$, where divergence between LT and ST starts at point $(10, 3)$ and $\vartheta = \pi/6$. Before diverging, the two trajectories are not exactly the same because of the cubic interpolation done in order to avoid singularities in later numerical derivations. Fig. 8.2b shows the same trajectories (only the second component of each 3D position vector) as function of time together with the corresponding GNSS measurements, where the velocity absolute value is constant.

8.5.1 Quadratic Programming

We evaluate the detection performance of the algorithm in Section 8.4.2 in terms of DET, i.e., false alarm and misdetection probabilities defined as follows

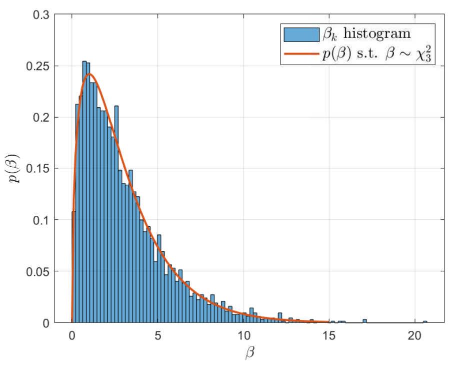
$$P_{\mathrm{FA}} = \mathbb{P}\left[\hat{\mathcal{H}} = \mathcal{H}_1|\mathcal{H}_0\right], \tag{8.58}$$

$$P_{\mathrm{MD}} = \mathbb{P}\left[\hat{\mathcal{H}} = \mathcal{H}_0|\mathcal{H}_1\right]. \tag{8.59}$$

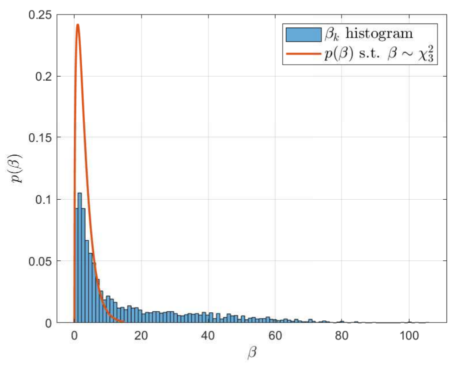
We estimate the DET curves with Montecarlo simulations and show the results in Fig. 8.3. The algorithm takes as input a window of measurements with duration $t_w = 1$ s and outputs a decision at time t_0. The window size has an impact on memory, since matrix A in Section 8.2.2 grows with the number of considered samples.

We first explore ϑ in Fig. 8.3a, were we can see that bigger ϑ yields better performance, confirming that our algorithm effectively captures the diversity of the LT and ST as it increases.

Fig. 8.3b shows how the choice of t_0, given a fixed t_w, impacts DET curves. In our scenario divergence between LT and ST starts at time $t_d = 5$ s as shown in Fig. 8.2b. For $t_0 = t_d + t_w$ (blue line in Fig. 8.3b) the two trajectories are not distinguishable because measurement noise hides the trajectory divergence, and indeed the corresponding DET curve is almost the trivial one (a diagonal in the DET plot with linear axes). For $t_0 > t_d + t_w$ (red, yellow and violet lines), instead, performance increases significantly.



(a) Histogram plot of $\beta_k|\mathcal{H}_0$.



(b)

Figure 8.4: Histogram plots of β_k under a LT and ST, i.e., under $\mathcal{H}_0$ and $\mathcal{H}_1$.

8.5.2 Innovation Testing

We have implemented [137, Algorithm 3 and Appendix B], i.e., an EKF that does pose and orientation estimation via sensor fusion. The algorithm takes as input GNSS and IMU (accelerometer and gyroscope) measurements, and defines as state at iteration t

$$\boldsymbol{x}_t = [\boldsymbol{p}_t, \boldsymbol{v}_t, \boldsymbol{q}_t^{(\mathrm{nb})}]^t, \tag{8.60}$$

where $\boldsymbol{v}_t$ is the velocity in the navigation frame. The measurement process is given instead by the GNSS measurements.

We run Montecarlo simulations on a LT and ST and collect the realizations of the innovation test statistic (8.56) and estimates its PDF $\hat{p}(\beta)$. Results are shown in Fig. 8.4.

We first apply the EKF on a LT. Under $\mathcal{H}_0$ we expect that β is Chi-squared distributed with 3 degrees of freedom, since the measurement vector of the EKF is 3-dimensional. Hence we have

$$\beta \sim \chi_3^2, \quad p(\beta|\mathcal{H}_0) = \frac{1}{2^{k/2}\Gamma(k/2)}\beta^{k/2-1}e^{-\beta/2}, \tag{8.61}$$

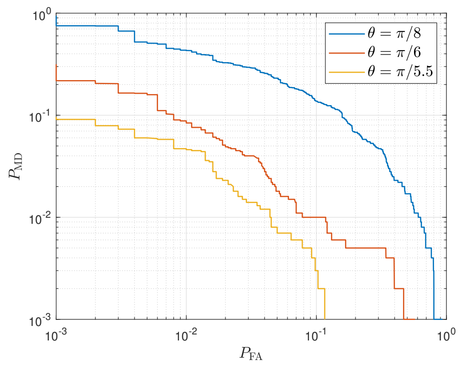
where $\Gamma(\cdot)$ is the well-known gamma function. Fig. 8.4 shows $\hat{p}(\beta_k|\mathcal{H}_0)$ together with $p(\beta|\mathcal{H}_0)$ and we can see how the two distribution match, as expected.

Then we apply the EKF on a ST and Fig. 8.4b shows how in this case the innovation test is not chi-squared distributed.

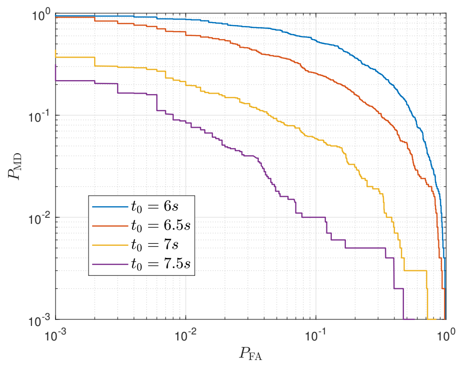
In order to compare the two anti-spoofing algorithms against a common scenario and the same parameters, we set 100 Hz as the IMU measurement rate and 10 Hz as the GNSS measurement rate, as done for the previous algorithm. As in Section 8.5.1 a window of 1 second worth of innovation values was used for detection purposes, with varying t_0 (test instant) and ϑ for the trajectory. The DET curves were derived through Montecarlo simulations, by collecting the statistics of the normalized innovation at different time instants, both in the authentic and spoofing case. The results are reported in Fig. 8.5.

8.5.3 Performance comparison

Fig. 8.6 and 8.7 highlight the comparison between quadratic the anti-spoofing method based on quadratic programming and innovation testing based on the EKF. As it can be

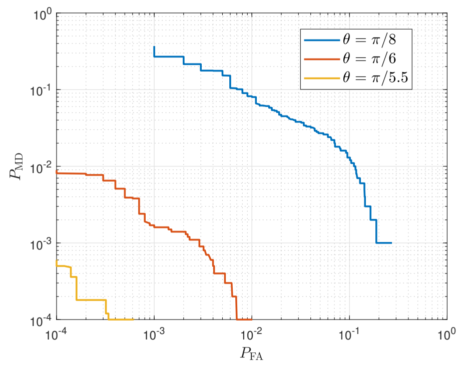


(a) DET curves for different ϑ ($t_0 = 7.5$s).

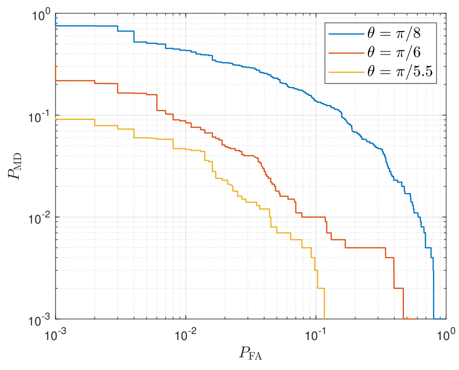


(b) DET curves for different t_0 ($\vartheta = \pi/6$)

Figure 8.5: DET curves for spoofing detection with the EKF.



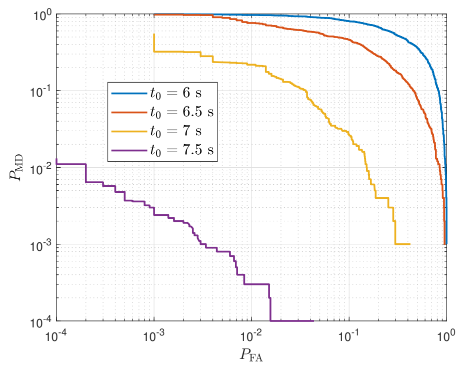
(a) DET curves for different ϑ with quadratic programming.



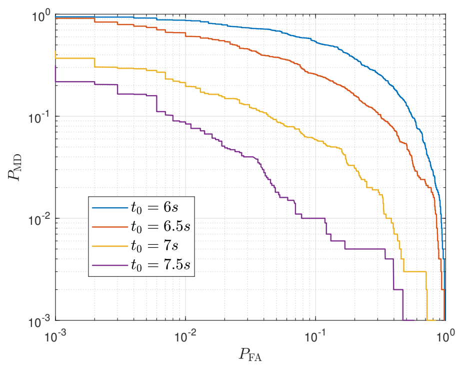
(b) DET curves for different ϑ with EKF.

Figure 8.6: DET curves for spoofing detection, varying ϑ.

noticed, the quadratic programming method on an observation window of one second achieves better performance. The difference in performance is enhanced for certain parameter values. As an example, when the difference between the two trajectories is more marked, the quadratic programming method achieves considerably better performances.



(a) DET curves for different t_0 with quadratic programming.



(b) DET curves for different t_0 with EKF.

Figure 8.7: DET curves for spoofing detection, varying t_0.

Chapter 9

Resilient Distance Bounding for Cellular Networks

The upcoming 5G standard will bring a major boost in telecommunication networks. Future 5G networks are planned to offer significantly higher bandwidth at higher carrier frequencies [145]. This choice comes together with the challenge of more demanding propagation environments, thus network densification will also play an important role. The ubiquity of densely deployed access nodes will favor the widespread of new services and features that are envisioned to become deeply integrated with everyday life. One of these features is location awareness.

With the 5G scenario in mind, we focus on the problem of authenticating the user's position to network authorities. The proposed authentication protocol will exploit reasonable 5G network and hardware features and draw from solutions proposed in the scope of GNSS.

9.1 User authentication

This work focuses on authenticating the user's position to the network, considered in this preliminary phase as a unique, time synchronized entity. Two are the main target threats the authentication mechanism should overcome: external attackers that aim at falsifying another user's position or internal attackers that aim at forging their own position. Internal attacks, or self spoofing, are harder to defeat, as the attacker coincides with the legitimate user and has incentive in breaking the protocol rules. For this reason the proposed protocol is network based rather than user based, in order to leave less freedom to users that may be willing to cheat the system. Each base station will broadcast an authentication signal containing unpredictable symbols; the user will receive the superposition of these signals, sign it and send it back to the network authority. The attacker is assumed to possess unlimited resources in terms of processing power and distributed high gain directional antennas. This implies the attacker is capable of replaying a delayed version of the authentication signals sent by each base station. With these assumptions, the only constraint that prevents an internal attacker from authenticating a forged position is geometry and time synchronization. Indeed, as seen in chapter 3 in order to forge the authentication signals for an arbitrary position at the same time, some of the signal will need to be delayed and some would need to be anticipated. Since the transmitted signals are unpredictable, however, anticipating them without adding any time delay forces the attacker to perform random guessing. As seen in 3, where the attacker was forced to estimate the unpredictable symbols, the effect of this zero-delay estimation are easier to observe when the symbol rate is high (shorter symbols) and when the detection is based on a large number of unpredictable symbols. With these considerations, the authentication protocol concept stirs towards a *distance bounding* approach.

9.1.1 Distance bounding

Distance bounding is a category of security protocols that aim at authenticating a lower bound for the distance between prover and verifier, making it impossible for the former to appear closer than its actual distance, [146, 147]. Distance bounding protocols estimate a bound to the prover-verifier distance through measurements of the transmission time. The processing time between the reception of the challenge and the transmission of the response is minimized, so that the round trip time is mostly due to the actual propagation time.

The most relevant threats to a distance bounding protocol are the following [148]:

- *Mafia attack*: in this man-in-the-middle attack the adversary Eve interferes with the communication between the honest prover Bob and the verifier in order to make Bob appear closer.
- *Distance attack*: a dishonest prover attempts to trick the verifier into believing he is within a certain distance threshold, while he is not.
- *Terrorist attack*: a dishonest prover colludes with the adversary Eve in order to help her pass the protocol successfully. It is argued in [149] that the distance bounding literature is divided on the kind of information the prover can share with the attacker. The most empowering definition for the attacker allows the prover to share anything that is more useful to Eve in the current protocol phase than in the following. The key point in the definition is that whenever the prover stops collaborating, the attackers success probability is negligibly higher that that of the Mafia attack.

Distance bounding was first introduced by Brands and Chaum in [150] with the aim of detecting the mafia fraud. Most distance bounding protocols are composed by different phases, that can be either *lazy* (i.e., no round trip time measurement) or *time critical* (i.e., the clock is used to determine the round trip time). In the latter case the latency must be minimized as the challenge-response time is used to bound the user's distance [149]. An important requirement in distance bounding is the commitment by each of the entities to a previously agreed value that is communicated in an encrypted fashion *before* the time critical phase and disclosed *after* it. This protects from external attackers, ensuring that only a response computed with the rightful credentials is accepted by the verifier. The targets of a distance bounding protocol are the following:

1. the system shall never accept a reply computed with right credentials from a device that is further than the distance threshold;

2. the system shall accept a reply computed with the right credentials from a device that is placed at any distance within the threshold.

3. the system shall never accept a reply computed without credentials;

4. no information on the credentials shall be derived from the challenge response exchange;

Distance bounding protocols are usually designed for resource constrained scenarios such as RFID tags; adapting a distance bounding protocol for cellular network scenarios is still an open problem. The authors of [151] investigate a secure position authentication scheme for a generic wireless network, in which the base stations need to authenticate the location of a mobile device. The proposed mechanism is called *verifiable multilateration* and it extends the concept of distance bounding [150], to 3D authenticated positioning. With the collaboration of at least four base stations, the position of the user can be authenticated: provided the base stations are laid out with a good geometry at least one of the distances will have to be shortened in order for the attacker to fake its position. In [152] the verifiable multilateration protocol proposed in [151] was implemented in a realistic scenario. However the authors claim that an attacker exploiting distributed antennas in proximity of each base station, where each antenna owns a copy of authentication credentials, can easily get any fake position to pass the authentication test.

Starting from this work, we devise a different protocol that is resilient against a more sophisticated adversary.

9.2 Concurrent distance bounding protocol

This work proposes to extend the position authentication protocol presented in [151] by involving all the nearby base stations simultaneously (instead of iterating the protocol one base station at a time). The devised position authentication algorithm does not have to be concurrent with the positioning protocol itself, but can be performed *a posteriori*. Therefore we split the protocol in a positioning phase, where the network infers the user's position, or alternatively the user communicates its claimed position to the network, and an authentication phase, where the network verifies its authenticity.

Let us start by making a simple consideration regarding the cellular networks scenario. If the security target in classic distance bounding scenarios are the ones in Section 9.1.1 and relate to a single range measurement, the adaptation to cellular networks may allow to enhance the security target, thanks to the availability of several range measurements from multiple base stations.

The probability that the system accepts a reply computed with correct credentials shall tend to 1 as the claimed position approaches the actual one, while the same probability shall vanish as the distance between the actual and claimed positions increases. In particular:

1. the system shall accept with probability at least $1 - \delta$ a reply from a device that is less than d_{near} away from its claimed position.

2. the system shall accept with probability at most ε a reply from a device that is further than d_{far} away from its claimed position, where $\delta, \varepsilon, d_{near}, d_{far}$ are parameters of the mechanism design.

By extending the security targets we can also consider a wider set of threats, including for example the *distributed terrorist attack*: a dishonest prover, Bob, colludes with several entities by sharing his own credentials in order for them to be able to authenticate a different position in place of Bob's true position. Notice that if none of these colluding entities is placed in the claimed position, then this attack would violate the revised version of security target number 1, but not the old version (in Section 9.1.1). The protocol described in [151] is vulnerable to this attack. If Bob places an antenna with cloned credentials in close proximity to each of the base stations, the iterative distance bounding protocol described in [151] will accept each of the distributed antenna's response, as they are, indeed, closer than the claimed position. This will result in Bob being able to authenticate any position by simply setting up the distributed antenna system and operating it remotely. The aim of this work is to propose a protocol that can overcome this kind of attack.

Instead of performing N_{BS} independent iterations of the classic distance bounding, the protocol will involve all base station in a single concurrent challenge transmission. The response should be a function of the superposition of all N_{BS} symbols, signed with a previously agreed key. The distance bounding sequence transmission will be synchronized *at the user claimed position*, by exploiting the knowledge on the position that is to be authenticated and the time synchronization between base stations. This means that each base station B_i will transmit its own symbol s_i with a time delay $\Delta t_i = \frac{d_{u \to B_i} - d_{\text{min}}}{c}$, where $d_{\text{min}} = \min_i\{d_{u \to B_i}\}$ and $d_{u \to B_i}$ is the distance between the ith base station and the user's position. This ensures that all symbols will be received concurrently by a honest receiver, who will engage in the distance bounding protocol by responding to the *superposition* of the challenge sequence.

This approach allows to mitigate the threat of the distributed terrorist attack. Indeed now the position authentication protocol is tailored to the position that is to be authenticated and no antenna distribution allows the attacker to gain time advantage for symbol replay. Moreover this slotted communication synchronized at the receiver's side is well suited for cellular network technologies such as 3GPP LTE and NR, which have a rigid and synchronized frame structure [145].

One may argue that if an attacker can manage to have distributed antennas that can authenticate to the network in place of the user itself, then it might as well position a clone device in the position it wants to fake and have it answer in place of the real device. There are two relevant consideration regarding this attack:

1. in case of the distributed antennas, it was argued in [151] that such a setup for the attacker enables it to spoof *any possible position* in the nearby area. On the contrary this attack forces the attacker to move the clone device along with the fake position, increasing the attack complexity.

2. if the attacker can actually manage to place a clone device in the false position then the protocol *should* accept the response, by the revised version of security target

number 2.

In light of the above considerations we conclude that a protocol that is resilient against distributed terrorist attack is more in line with the security target of a distance bounding protocol tailored to cellular scenarios.

9.3 System Model

The aim of this proposal is to authenticate the user position to the network, relying on the cooperation of multiple base stations. As the purpose of this analysis is mainly to assess the achievable level of security of the authentication protocol, the details of the adaptation to cellular networks are not discussed for the time being. The parties that are involved in the authentication scheme are:

- **Network:** we consider B base stations, connected and time-synchronized with each other. The base stations can transmit directionally, i.e., they are equipped with beamformers, allowing to limit the transmission to a sector of angular width α. The transmissions are slotted and the slot duration is T. The base stations are all synchronized with the beginning of the slot at the receiver's side, i.e., at the beginning of time slot N_T, t_{N_T}, the user will receive the superposition of B symbols. Each of the symbols is Gaussian with zero mean and standard deviation σ_s, differently from classic distance bounding implementations, that usually exploit binary symbols [151]. Each of the time slots will fit N_s symbols of duration $\frac{T}{N_s}$.
- **User:** it can be equipped with (i) an omnidirectional transceiver, which is a simple device, capable of transmitting and receiving from multiple directions, but does not provide beamforming gain; or (ii) a transceiver capable of hybrid or digital beamforming with M RF chains [153], i.e., capable of transmitting and/or receiving from multiple directions with beamforming gain, but complex (at least for transmission) and with a higher energy consumption. We assume that for the beamforming solution (ii) the user is capable of transmitting its payload back to K base stations, where $K \leq M$. In case of omnidirectional transmission, (i), the user is capable of sending its response back to each of the B base stations. We also assume the user is able to sample the channel at a high enough frequency to have a sufficient granularity of the received signal. The user will receive the superposition of the distance bounding symbols and send back the resulting superposition symbol right after reception, with a maximum latency of L, mainly due to time synchronization errors and hardware delay.

9.4 Proposed Authentication Scheme

The steps of the authentication protocol are described in the following:

- The user communicates to the network the position it wants to authenticate, that will be referred to as the *claimed position*. This is derived during a previous step that may either involve the network or an alternative positioning system.

- The base stations coordinate to transmit the distance bounding challenge in downlink. In particular, as discussed in Section 9.1.1, the base stations will transmit according to their distance to the claimed position, so that the challenge symbols superimpose at the receiver's side. Therefore, the base station B_i transmits its random symbol s_i at time $t_0 + \Delta t_i$, where

$$\Delta t_i = \frac{d_{u\to B_i} - d_{\text{min}}}{c}, \tag{9.1}$$

  where $d_{\text{min}} = \min_i \{d_{u\to B_i}\}$, and $d_{u\to B_i}$ is the distance between the ith base station and the user's position.

- The user receives the superposition of the distance bounding challenges, applies a non-linear transformation and replies to all the base stations that joined the distance bounding procedure. The non-linear transformation must depend on the authentication credentials in a pseudorandom manner.

- Each base station B_i measures the round-trip time, Ω_i, and estimates the user range $\hat{d}_{u\to B_i}$. If this does not match the expected range (i.e., $\hat{d}_{u\to B_i} \simeq d_{u\to B_i}$), then the authentication fails.

9.4.1 Threats and Attacks

Let us consider an omni-potent attacker with infinite resources (i.e., it may have an arbitrary number of distributed or non-distributed directional antennas). In this network authentication scenarios there are two relevant types of attacks:

- **External-spoofing attack**, in which a malicious attacker aims at falsifying the position of a legitimate user. As an external attacker does not own authentication credentials, the only feasible attack is the replay of the downlink authentication signals. This attack is similar to the mafia attack.

- **Self-spoofing attack**[1], in which a malicious user tries to trick the system into thinking he is in a different position. This kind of attack is the harder to detect and prevent, since the user, who owns authentication credentials, plays against the protocol itself. As reported in the previous sections, the attacker may exploit a network of distributed antennas (possibly carrying the cloned authentication credentials) in the attempt of modifying the downlink symbol superposition to authenticate an arbitrary position. This attack is analogous to the distance fraud in distance bounding protocols.

The self spoofing attack is a more powerful version of the external attack in which the adversary owns the authentication credentials. Therefore we can focus solely on the latter.

[1]It is worth noticing that self-spoofing in this specific cellular network scenario is a different concept with respect to self-spoofing in the GNSS scenario. Whereas the latter is performed by the attacker on its own receiver, without involving other system entities, the former ultimately aims at tampering with the reported position at the base stations side. The target is therefore more immediately identified with external network entities.

The self-spoofer will perform its attack by claiming to be in any arbitrary position and placing its distributed antennas in the most favorable position. The best choice for the attacker is to place B antennas in proximity of each of the involved base stations in the direction that connects each of them with the claimed position. The antennas can be assumed to have a copy of the authentication credentials. This attack is the most powerful among those considered in literature for distance bounding, as it goes beyond the terrorist attack, where only short term credentials can be shared with other rogue devices.

The attack can adopt different strategies in order to deceive the system and have its fake claimed position pass the verification test. Notice that even when exploiting the distributed antennas with credentials, the protocol requires the attacker to collect the distance bounding symbols from each of the involved base stations before being able to compute the response. This is guaranteed by the non linearity of the signature operation, that ensures the response cannot be decomposed into B terms that can be computed separately by each antenna and then transmitted so that the superposition at each base station gives the correct response. For this reason the attacker is forced to introduce a delay in the downlink phase, waiting until all symbols are collected. Let us consider two attack strategies:

1. collect all the B symbols and transmit the response coherently with the delays that would be experienced from the claimed position, i.e., communicate independently with each base station with beam forming and delay each of the responses according to the distance from the base station to the claimed position.

2. collect only part of the B symbols and compute the response on a partial superposition.

9.4.2 Channel model

As an initial ideal scenario, we consider an AWGN channel model with noise power σ_w^2 and propagation delay $\tau(d)$. An extension to this model should take into account the channel gain, propagation loss, fading and shadowing.

9.4.3 Superimposed distance bounding symbols

In general distance bounding protocols are designed to minimize the processing time required between reception and transmission of the reply, therefore they mostly employ very basic operations (e.g., XOR) on binary symbols. However, since our protocol is designed to compute the reply on the superposition of symbols from several entities, we must consider the effect of such superposition on the final symbol distribution.

Let us make the assumption that the distance bounding signals are analog, Gaussian symbols with zero mean and variance σ_s^2. If the symbol duration is $T_s = \frac{T}{N_s}$, we assume the sampling frequency is high enough to capture the whole bandwidth of the distance bounding sequences, that is $F_s \geq 2\frac{1}{T_s}$. The users will see each of the N_s symbols as a random variable, deriving from the superposition of B i.i.d. Gaussian symbols, while the system can be assumed to have perfect knowledge of all the sequences. At the prover's receiver the nth distance bounding symbol will be:

$$s_n^R = \sum_{i=1}^{B} s_i^n + w_n^R, \;\; w_n^R \sim \mathcal{N}(0, \sigma_R) \tag{9.2}$$

where w_n^R represents the Gaussian noise at the receiver's side in the AWGN model and s_i^n is the nth symbol transmitted by the ith base station. The probability density function of s_n^R given the value of the transmitted symbols is:

$$p(s_n^R | s_n^1, \dots, s_n^B) = \frac{1}{\sigma_R \sqrt{2\pi}} e^{-\frac{(s_R^n - \sum_{i=1}^{B} s_i^n)^2}{2\sigma_R^2}} \tag{9.3}$$

In order for the protocol not to be vulnerable to distributed antenna attacks the reply computed by the user must be a non-linear operation in the key bits. We assume each user is equipped with a unique, previously agreed full-entropy binary sequence. In the proposed protocol the user will perform the following non-linear operation, that is an exclusive or on the sign of the received symbol:

$$r^n = |s_n^R| \left(\text{sign}(s_n^R) \oplus k_n\right) \tag{9.4}$$

where k_n is the nth bit of the secret sequence, taking values in $[-1, 1]$. Notice that both k^n and s_n^R determine the sign of the response symbol. The receiver's response is sent to each base station and takes the form:

$$s_n^{\text{bs},i} = r_n + w_i^B, \;\; w_i^B \sim \mathcal{N}(0, \sigma_i) \tag{9.5}$$

The distribution of the symbol received by each base station is:

$$s_{\text{bs},i}^n \sim \mathcal{N}\left(\left| \sum_{i=1}^{B} s_i^n \right| \left[\text{sign}\left(\sum_{i=1}^{B} s_i^n \right) \oplus k_n \right], \sigma_{\text{bs},i} \right) \tag{9.6}$$

with $\sigma_{\text{bs},i}^2 = B\sigma_s^2 + \sigma_R^2 + \sigma_i^2$. Notice that despite the non-linear operation at the receiver's side the distribution of the symbol received at each base station, given the values of the transmitted symbols, is Gaussian. The observation at each base station i is a vector of symbols starting at time $t_{0,i}$, the expected time of arrival of the first response symbol, long enough to collect all symbols of the sequence. In this first analysis we assume that the collected observation vector $\mathbf{s}_{\text{bs},i}$ is exactly synchronized with the symbols of the sequence.

The attack hypothesis $\mathcal{H}_1$ is linked to a spoofing event of unknown type. As discussed the attacker has two main strategies, that are:

- collect all B symbols and send the response separately to each base station, maintaining range coherence with respect to the spoofed positions;
- collect only part of the B symbols, in the attempt of minimizing the introduced delay at the cost of success probability.

In any of these cases the attack will perturb the observation vector, either shifting the values in time or changing the distribution of the received sequence. The time shifting can be detected by exploiting time synchronization at the system side and geometric considerations enforced by the distance bounding. Bounds on the time delay introduced by the attacker are derived in section 9.5.1. On the other hand it is also relevant to discuss the detection performances in case the attacker tries to decrease the introduced time delay by integrating only a subset of the symbols in its reply. This assessment is performed through binary hypothesis testing, also in section 9.5.2.

9.5 Security Assessment

9.5.1 Time bias introduced by the attacker

Let us consider the *self spoofing attack*, that is the most challenging for the network, as the receiver is willing to deceive the system and play against protocol rules.

We consider an attack effective if a device that possesses authentication information manages to successfully authenticate a position that is more than α meters away from the one where it is actually located. The tolerance α must be defined in order to account for several errors that may derive from:

- $\delta_{\hat{p}}$, which is the error that affects the estimate of the claimed position;
- δ_t which is the range error due to the maximum detectable propagation delay;
- δ_{proc} which is the maximum processing delay that the system is willing to tolerate at the receiver's side.

Let us consider the scenario represented in Fig. 9.1. We consider the attack strategy number 1 discussed in Section 9.4.1 and evaluate the time budget of the attacker in the round trip time with respect to that of an authentic protocol session. We are interested in finding the value of $\Delta^{\text{RTT}}_{a,u}$, i.e., the minimum delay the attacker is forced to introduce when carrying out this kind of attack. We expect that this value will depend on the relative distance between the attackers position and the claimed one, and that it will be influenced by the geometry of the involved base stations. We recall that $d_{x\to y}$ represents the distance between elements at positions x and y in the networks, and we call a the attacker's position, u the user position to be authenticated and B_i for i-th base station. Let us define:

$$B^u_{\text{min}} := \arg\min_{B_i}\{d_{u\to B_i}\}; \quad B^{u,a}_{\text{max}} := \arg\max_{B_i} \frac{d_{a\to B_i} - d_{u\to B_i}}{c} = \arg\max_{B_i} \Delta t^{B_i}_{u,a}; \tag{9.7}$$

that are the base station at minimum distance with respect to the claimed position and the base station for which the difference in the propagation time to the attacker's position and the claimed position is the maximum. B^u_{min} plays an important role in bounding the minimum delay the attacker has to introduce in downlink.

Downlink When considering the downlink i, we refer to the time interval D^i_x between the start of transmission at base station i and the start of the uplink communication from

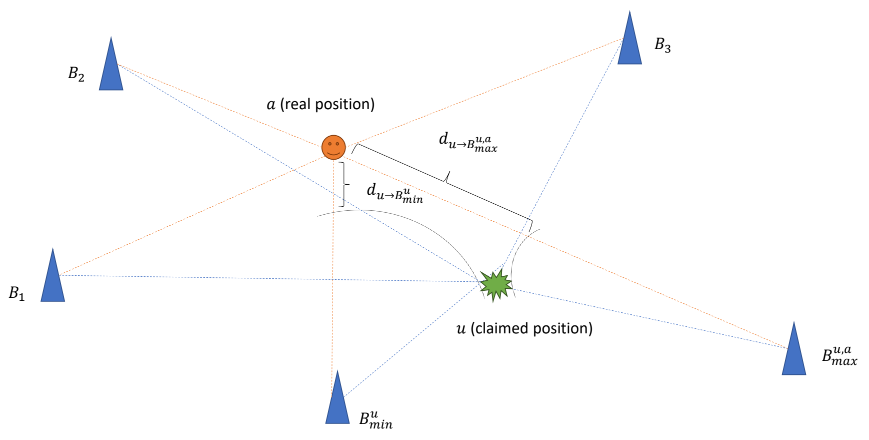


Figure 9.1: Scheme of the considered self spoofing scenario.

position x. We assume the attacker needs to wait until it receives all the symbols from each involved base station. By construction, the symbols will arrive at different time instants in any position that is different from the claimed one. Let us call D^i_a and D^i_u the downlink time for B_i for the authentic position u and for the attacker's position a, respectively. Then we have that at the claimed position:

$$D^i_u = \frac{d_{u \to B_i}}{c} + \frac{T}{N_s} + t^u_{\text{proc}} \tag{9.8}$$

where t^u_{proc} is the processing time for the computation of the reply for the legitimate device. Correspondingly at the attacker's position:

$$D^i_a = \frac{d_{u \to B_i}}{c} + \Delta t^{B_{\text{max}}}_{u,a} + \frac{T}{N_s} + t^a_{\text{proc}} \tag{9.9}$$

Therefore the difference between the downlink time with respect to the legitimate position is:

$$\Delta^D_{a,u} = \Delta t^{B_{\text{max}}}_{u,a} + (t^a_{\text{proc}} - t^u_{\text{proc}}) \tag{9.10}$$

Notice that the introduced delay is equal for all base stations. This is due to the fact that $B^{u,a}_{\text{max}}$ is setting the waiting time for the attacker, as it is the last base station from which the symbol will be received.

Uplink The attacker will transmit a different uplink to each base station, mimicking the relative delays they would experience if the reply actually came from the claimed position. Let us derive the delay the attacker has to impose on each uplink transmission. Calling $t^{B_i}_u := \frac{d_{u \to B_i}}{c}$, and $t^{B_i}_a := \frac{d_{a \to B_i}}{c}$, we argue that ideally the attacker will apply the delays $\Delta t^i_{\text{uplink}}$ such that:

$$t^{B_i}_a + \Delta t^i_{\text{uplink}} = t^{B_i}_u \tag{9.11}$$

And the delays can be computed as:

$$\Delta t^i_{\text{uplink}} = t^{B_i}_u - t^{B_i}_a = -\Delta t^{B_i}_{u,a} \tag{9.12}$$

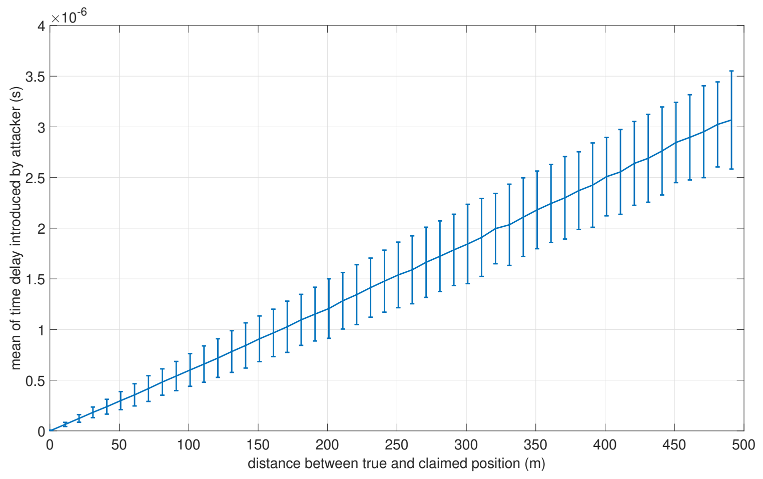


Figure 9.2: Monte Carlo simulation of the delay introduced by the attacker due to geometry.

However in general some of these delays will be negative. As the attacker can not go back in time, it will have to wait for a time equal to:

$$\Delta t^*_{\text{uplink}} = \left| \min_i \{\Delta t^i_{\text{uplink}} : \Delta t^i_{\text{uplink}} < 0\} \right| = \Delta t^{B_{\max}}_{u,a} \tag{9.13}$$

Therefore the propagation times will be:

$$U^i_a = t^{B_i}_a + \Delta t^i_{\text{uplink}} + \Delta t^{B_{\max}}_{u,a} = t^{B_i}_u + \Delta t^{B_{\max}}_{u,a} \tag{9.14}$$

while in the legitimate position it would be:

$$U^i_u = t^{B_i}_u \tag{9.15}$$

Therefore the imposed uplink delay, which again is applied to all ranges, is:

$$\Delta^U_{a,u} = \Delta t^{B_{\max}}_{u,a} \tag{9.16}$$

By putting together all the results we obtain the overall round trip time delay, imposed to all ranges:

$$\Delta^{\text{RTT}}_{a,u} = \Delta^U_{a,u} + \Delta^D_{a,u} = 2\Delta t^{B_{\max}}_{u,a} + (t^a_{\text{proc}} - t^u_{\text{proc}}) \tag{9.17}$$

We perform a rough assessment of the Round trip time delay introduced by an attacker with respect to the distance between the true and claimed positions through Monte Carlo simulations. The processing time was set to zero for both the attacker and the legitimate users, while the base stations were deployed following a Poisson Point Process with a density of 10 base stations per square km. Therefore the results, reported in Fig. 9.2, depend only on geometry.

9.5.2 Sequence tampering detection

In this section we evaluate the detection performances of an attack that exploits only a subset of the symbols involved in the protocol to compute the reply. The rationale behind this evaluation is that in order to avoid introducing a big time delay and being detected, the attacker may have a better chance by excluding a few base station sequences from its reply. The excluded base stations will be those that are most unfavorable for the attacker from a geometric point of view, in that they force it to introduce a higher time delay. In this evaluation we consider the effect of a forged reply calculated on a proper subset of the involved base station sequences. Without loss of generality we assume that the forged sequences arrive at the base station side with consistent time delays. This sections evaluates the detection capabilities of the system when one or more sequences are deliberately omitted from the response computation.

Let us call $\hat{\mathcal{B}}$ the set of base station sequences the attacker observes, where $\hat{\mathcal{B}} \subset \mathcal{B}$ and $\mathcal{B}$ is the set of the sequences from all the involved base stations. Following the considerations in section 9.4.3 we derive the two hypotheses whose likelihood the system needs to evaluate to detect a spoofing attack:

$$\mathcal{H}_0 : \mathbf{s}_{\mathrm{bs},i} \sim \mathcal{N}\left(\left|\sum_{i=1}^{B} \mathbf{s}_i\right| \left[\operatorname{sign}\left(\sum_{i=1}^{B} \mathbf{s}_i\right) \oplus \mathbf{k}\right], \mathbf{\Sigma}_{\mathrm{bs},i}\right) \tag{9.18}$$

$$\mathcal{H}_1 : \mathbf{s}_{\mathrm{bs},i} \sim \mathcal{N}\left(\left|\sum_{i\in\hat{\mathcal{B}}} \mathbf{s}_i\right| \left[\operatorname{sign}\left(\sum_{i\in\hat{\mathcal{B}}} \mathbf{s}_i\right) \oplus \mathbf{k}\right], \hat{\mathbf{\Sigma}}_{\mathrm{bs},i}\right) \tag{9.19}$$

where $\mathbf{\Sigma}_{\mathrm{bs},i}$ and $\hat{\mathbf{\Sigma}}_{\mathrm{bs},i}$ are two diagonal matrices since both the noise and the symbols are uncorrelated in time, and the elements on the diagonal correspond to $\sigma^2_{\mathrm{bs},i} = B\sigma_s^2 + \sigma_R^2 + \sigma_i^2$ and $\hat{\sigma}^2_{\mathrm{bs},i} = |\hat{\mathcal{B}}|\sigma_s^2 + \sigma_R^2 + \sigma_i^2$. The $|\cdot|$ operation on a vector is intended here as the element-wise absolute value.

The authenticity test is performed through binary hypothesis testing (BHT) by taking the logarithm of the ratio between the probability density function of the observation vector in the two hypotheses:

$$\Lambda = \log \frac{p(\mathbf{s}_{\mathrm{bs},i}; \mathcal{H}_0 | \mathbf{s}_1, \ldots, \mathbf{s}_B)}{p(\mathbf{s}_{\mathrm{bs},i}; \mathcal{H}_1 | \mathbf{s}_1, \ldots, \mathbf{s}_B)} \tag{9.20}$$

where we are making the preliminary assumption that the system knows the exact subset of sequences targeted by the attacker. In order to evaluate an upper bound for the detection statistics we evaluate the Kullback-Leibler divergence between the two distributions. This value indicates the statistical distance (or better, divergence, as it is not symmetric) between the two distributions. In the case of two multivariate Gaussian

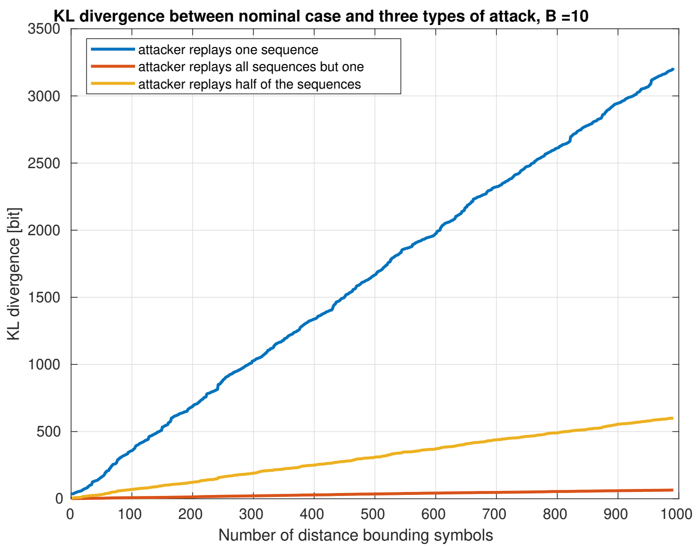


Figure 9.3: $D_{\mathrm{KL},i}(\mathcal{N}_0|\mathcal{N}_1)$ for $B = 10$ and $\sigma_r = 0.1581$ (corresponding to an SNR of 20 dB).

distributions the KL divergence assumes the form:

$$D_{\mathrm{KL},i}(\mathcal{N}_0|\mathcal{N}_1) = \frac{1}{2}\left(\mathrm{tr}\left(\hat{\boldsymbol{\Sigma}}_{\mathrm{bs},i}^{-1}\boldsymbol{\Sigma}_{\mathrm{bs},i}\right) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\,\hat{\boldsymbol{\Sigma}}_{\mathrm{bs},i}^{-1}\,(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - N + \ln\frac{|\hat{\boldsymbol{\Sigma}}_{\mathrm{bs},i}|}{|\boldsymbol{\Sigma}_{\mathrm{bs},i}|}\right) \tag{9.21}$$

$$= \frac{1}{2}\left(N\frac{\sigma^2_{\mathrm{bs},i}}{\hat{\sigma}^2_{\mathrm{bs},i}} + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\,\hat{\boldsymbol{\Sigma}}_{\mathrm{bs},i}^{-1}\,(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - N + N\ln\frac{\hat{\sigma}^2_{\mathrm{bs},i}}{\sigma^2_{\mathrm{bs},i}}\right) \tag{9.22}$$

$$= \frac{1}{2}\left(N\left(\frac{\sigma^2_{\mathrm{bs},i}}{\hat{\sigma}^2_{\mathrm{bs},i}} + \ln\frac{\hat{\sigma}^2_{\mathrm{bs},i}}{\sigma^2_{\mathrm{bs},i}} - 1\right) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\,\hat{\boldsymbol{\Sigma}}_{\mathrm{bs},i}^{-1}\,(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)\right) \tag{9.23}$$

This formulation is evaluated in Fig. 9.3 for different values of the sequence length N, and a fixed number of base stations $B = 10$ and a $\sigma_r = \sqrt{B\sigma_s^2/\Gamma} = 0.1581$, with $\sigma_s = 0.5$ and the SNR $\Gamma = 20$ dB. In particular, three different attacks were evaluated:

- the attacker only knows a single sequence out of B;
- the attacker knows half of the sequences (i.e., $\lceil B/2 \rceil$);
- the attacker knows all the sequences but one.

Notice that in this evaluation we are not accounting for the delays that the attacker incurs into when collecting and forwarding the sequences, as discussed in Sec. 9.5.1.

As expected, in Fig. 9.3 it can be seen that, if the attacker knows the all the sequences but one, then the divergence of the sequences used for the attack and the legitimate one is very small, for all values of N. Instead, the divergence is higher when considering the attack with half of the sequences and that with a single sequence. This suggests that in most geometries the scheme can distinguish a legitimate reply from a forged one.

Fig. 9.4 reports the divergence for different values of B, and a fixed $N = 100$ and SNR $\Gamma = 20$ dB. The attack in which the malicious user knows $B - 1$ sequences improves as B increases, given that the weight of the unknown sequence decreases as the number of

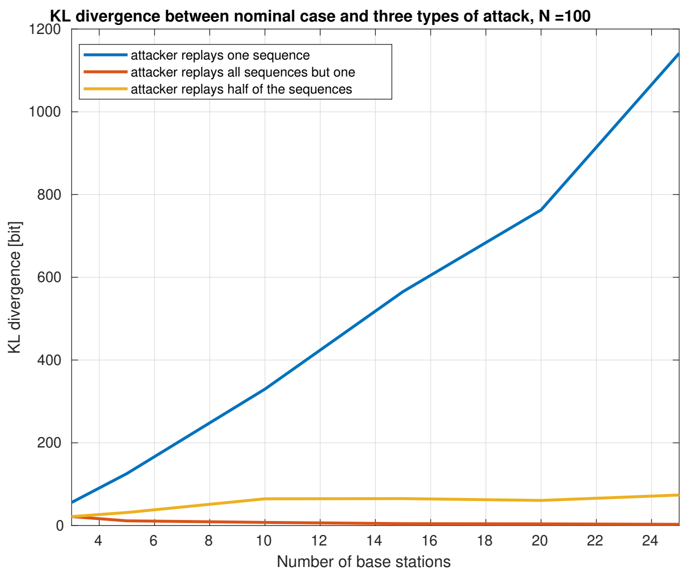


Figure 9.4: $D_{\mathrm{KL},i}(\mathcal{N}_0|\mathcal{N}_1)$ for $N = 100$ and an SNR of 20 dB.

sequences increase. However, notice that collecting an increasing number of sequences implies higher delays in the attack, which can be detected as reported in Sec. 9.5.1. The two other attacks, instead, exhibit an increasing divergence with B, even though after $B = 10$ the divergence of the attack that targets half of the sequences flattens.

Finally, Fig. 9.5 reports the performance as function of the SNR Γ of the receiver (of the legitimate user), while the receiver noise at the base stations and at the attacker do not change. Interestingly, it can be seen that with a smaller SNR the divergence is higher: this is probably due to the fact that the noise plays the role of an additional sequence, which is not known to the attacker. Notice that this does not imply that the proposed scheme works better in a high noise regime, given that other elements (e.g., the detectability of the sequence) may be affected by the low SNR.

Although derived independently, the proposed solutions has several similarities to the one presented in [154]. The authors underline the optimality of a simultaneous distance bounding protocol tailored to the claimer's position among all time-of-arrival based distance bounding solutions. Contrarily from [154], this protocol does not exploit bit sequences and frequency division among provers, but rather uses Gaussian symbols. An analysis on the difference in performance between these approaches is left for future work, together with an assessment on the effect of time misalignment among provers.

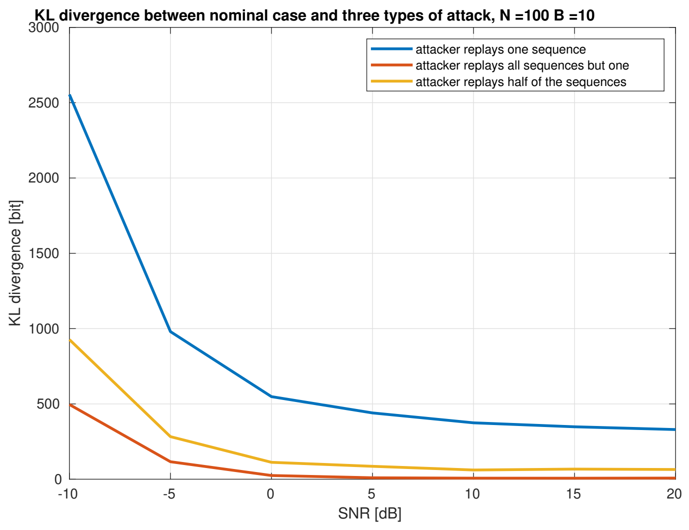


Figure 9.5: $D_{\mathrm{KL},i}(\mathcal{N}_0|\mathcal{N}_1)$ for $N = 100$ and $B = 10$.

Chapter 10

Conclusion

This thesis has investigated several aspects related to the security of GNSS, ranging from authentication to access control to attack detection and mitigation. As GNSS is composed of different domains, the same should hold for the adopted security countermeasures: protection mechanisms can be implemented at the data level, the signal level or at the PVT computation. Designing multiple security mechanism on different domains is beneficial, but the purpose of each countermeasure shall be well defined, pursuing separation between domains. As an example, while signal level countermeasures could in principle provide reasonable navigation data assurance, the opposite does not necessarily hold, as discussed in Chapter 3. Therefore in designing a protection mechanism the purpose and target should be unambiguous: data level techniques (e.g., digital signature) should aim at protecting the authenticity of the navigation message; signal level techniques should ensure the authenticity of the received signal, despite the distortion effects of the propagation environment, while PVT level techniques should provide some degree of confidence on the computed solution. In the choice and implementation of security algorithms it is important to remark that different applications have different requirements, security targets and resource availability. Some algorithms that work well for commercial grade applications may be too demanding for consumer grade receivers, as discussed in Chapter 7. Similarly, while the former may have higher security requirements, the latter will probably require protection against less sophisticated attacks. This justifies the investigation of numerous solutions, exploiting different tools and acting at different levels of the receiver architecture. The availability of a growing number of telecommunication systems, providing a wide range of the so called *signals of opportunity* is valuable for the implementation of cross-system security architectures. Moreover, the miniaturization of hardware components now guarantees the ubiquity of rich and diverse sensory information on most consumer devices, offering a precious source of position redundancy. With the upcoming fifth generation cellular networks, precise positioning will cease to be a GNSS prerogative. Therefore the investigation on position authentication should be carried on at different systems, possibly adapting solutions from one domain to the others. Chapter 9 tackles the problem of position authentication from a different perspective, exploiting a different approach (distance bounding) and investigating its adaptation to a cellular scenario. New tools that are worth exploiting sometimes come together with new threats that need to be mitigated. This is the case for quantum technologies, which on one hand cast a shadow on classic cryptography, but on the other hand offer a valuable tool to achieve unconditional security at the physical layer in wide range communications (inter-satellite and ground-satellite). This opportunity has been taken into consideration in Chapter 6, where key distribution protocols over a GNSS constellation are devised assuming the future implementation of inter-satellite quantum links.

However far from being an extensive overview of all security aspects related to GNSS, the effort of this thesis has focused on tackling the challenge of security GNSS from multiple points of view and with diverse tools. With the rapid growth of the market for precise and ubiquitous positioning services, the pool of feasible solutions worth exploring is envisioned to grow as well. Hopefully this work will be the first step to a broader investigation, which will ultimately result in a comprehensive architecture, integrating various security measures and increasing the resilience of GNSS and the reliability of PVT.

Bibliography

[1] F. Dovis, *GNSS Interference Threats and Countermeasures*, ser. Artech House GNSS technology and applications series. Artech House, 2015. [Online]. Available: https://books.google.com.au/books?id=XIcTBwAAQBAJ

[2] Chung-Liang Chang, "Novel multiplexing technique in anti-jamming GNSS receiver," in *Proceedings of the 2011 American Control Conference*. IEEE, jun 2011, pp. 3453–3458. [Online]. Available: http://ieeexplore.ieee.org/document/5990642/

[3] S. C. Lo and P. K. Enge, "Authenticating aviation augmentation system broadcasts," in *IEEE/ION Position, Location and Navigation Symposium*. IEEE, may 2010, pp. 708–717. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5507223

[4] J. T. Curran, M. Paonni, and J. Bishop, "Securing the Open-Service: A Candidate Navigation Message Authentication Scheme for Galileo E1 OS," in *European Navigation Conference, (ENC-GNSS)*, Rotterdam, 2014.

[5] I. Fernández-Hernández, "Method and system to optimise the authentication of radionavigation signals," *Patent EP14163902*, 2015. [Online]. Available: https://patentscope.wipo.int/search/en/detail.jsf?docId=WO2015154981

[6] P. Walker, V. Rijmen, I. Fernández-Hernández, G. Seco-Granados, J. Simón, J. D. Calle, and O. Pozzobon, "Galileo Open Service Authentication : A Complete Service Design and Provision Analysis," in *Proceedings of the 28th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2015)*, Tampa, Florida, 2015. [Online]. Available: https://www.ion.org/gnss/abstracts.cfm?paperID=2915

[7] I. Fernández-Hernández, V. Rijmen, G. Seco-Granados, J. Simón, I. Rodríguez, and J. D. Calle, "Design Drivers, Solutions and Robustness Assessment of Navigation Message Authentication for the Galileo Open Service," in *International Technical Meeting of The Satellite Division of the Institute of Navigation, ION GNSS*, 2014, pp. 2810–2827. [Online]. Available: http://www.ion.org/publications/abstract.cfm?jp=p{&}articleID=12532

[8] I. Fernández-Hernández, V. Rijmen, T. Ashur, P. Walker, G. Seco-Granados, J. Simon, C. Sarto, D. Burkey, and O. Pozzobon, "Galileo Navigation Message Authentication Specification for Signal-In-Space Testing - v 1.0," 2016.

[9] A. J. Kerns, K. D. Wesson, and T. E. Humphreys, "A blueprint for civil GPS navigation message authentication," in *2014 IEEE/ION Position, Location and Navigation Symposium - PLANS 2014*. IEEE, may 2014, pp. 262–269. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6851385

[10] K. D. Wesson, M. Rothlisberger, and T. E. Humphreys, "Practical Cryptographic Civil GPS Signal Authentication," *Journal of the Institute of Navigation*, pp. 1–15, 2011. [Online]. Available: http://www.ion.org/publications/abstract.cfm?jp=j{&}articleID=2576

[11] G. Caparra, S. Sturaro, N. Laurenti, C. Wullems, and R. T. Ioannides, "A Novel Navigation Message Authentication Scheme for GNSS Open Service," in *Proceedings of the 29th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2016)*, Portland, Oregon, 2016, pp. 2938 – 2947. [Online]. Available: https://www.ion.org/publications/abstract.cfm?jp=p{&}articleID=14692

[12] D. Naor, M. Naor, and J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers," *Advances in Cryptology CRYPTO 2001*, vol. 2139, no. June, pp. 1–36, 2001. [Online]. Available: http://link.springer.de//link/service/series/0558/papers/2139/21390041.pdf

[13] H.-S. Koo, O. H. Kwon, and S.-W. Ra, "An active entitlement key management for conditional access system on digital tv broadcasting network," in *2006 IEEE/IFIP Network Operations and Management Symposium NOMS 2006*, April 2006, pp. 1–4.

[14] H. M. Sun, S. Y. Chang, S. M. Chen, and C. C. Chiu, "An efficient rekeying scheme for multicast and broadcast (M&B) in mobile WiMAX," *Proceedings of the 3rd IEEE Asia-Pacific Services Computing Conference, APSCC 2008*, pp. 199–204, 2008.

[15] D. Boneh, C. Gentry, and B. Waters, "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys," *Advances in Cryptology – CRYPTO 2005*, vol. 3621, no. 1, pp. 258–275, 2005. [Online]. Available: http://link.springer.com/10.1007/11535218

[16] J. T. Curran and M. Paonni, "Securing GNSS: An End-to-End Feasibility Study for the Galileo Open Service," in *International Technical Meeting of the Satellite Division of The Institute of Navigation, ION GNSS*, 2014, pp. 1–15.

[17] L. Calderaro, C. Agnesi, D. Dequal, F. Vedovato, M. Schiavon, A. Santamato, V. Luceri, G. Bianco, G. Vallone, and P. Villoresi, "Towards quantum communication from global navigation satellite system," *Quantum Science and Technology*, vol. 4, no. 1, p. 015012, Dec. 2018. [Online]. Available: https://doi.org/10.1088/2058-9565/aaefd4

[18] F. Gerlin, N. Laurenti, G. Naletto, G. Vallone, P. Villoresi, L. Bonino, S. Mottini, and Z. Sodnik, "Design optimization for quantum communications in a gnss intersatellite network," in *2013 International Conference on Localization and GNSS (ICL-GNSS)*, June 2013, pp. 1–6.

[19] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979. [Online]. Available: http://doi.acm.org/10.1145/359168.359176

[20] T. Ebinuma, "GPS-SDR-SIM," https://github.com/osqzss/gps-sdr-sim.

[21] pmonta, "pmonta/GNSS-DSP-tools," available at https://github.com/pmonta/GNSS-DSP-tools.

[22] G. Caparra, S. Ceccato, N. Laurenti, and J. Cramer, "Feasibility and limitations of self-spoofing attacks on gnss signals with message authentication," 09 2017.

[23] G. Caparra, C. Wullems, S. Ceccato, S. Sturaro, N. Laurenti, O. Pozzobon, R. T. Ioannides, and M. Crisci, "Design Drivers for Navigation Message Authentication Schemes for GNSS Systems," *InsideGNSS*, vol. 11, no. 5, pp. 64–73, 2016. [Online]. Available: http://www.insidegnss.com/node/5101

[24] C. Wullems, O. Pozzobon, and K. Kubik, "Signal Authentication and Integrity Schemes for Next Generation Global Navigation Satellite Systems," in *European Navigation Conference, (ENC-GNSS)*, 2005, pp. 1–10.

[25] T. E. Humphreys, "Detection Strategy for Cryptographic GNSS Anti-Spoofing," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 2, pp. 1073–1090, apr 2013. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp={&}arnumber=6494400{&}contentType=Journals+{&}+Magazines{&}searchField=Search{_}All{&}queryText=gnss+security

[26] G. Caparra, N. Laurenti, R. T. Ioannides, and M. Crisci, "Improving Secure Code Estimation and Replay Attack and Detection on GNSS Signals," in *ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing, NAVITEC*, 2014.

[27] J. T. Curran, M. Bavaro, P. Closas, M. Anghileri, M. Navarro, B. Schotsch, and S. Pfletschinger, "On the Threat of Systematic Jamming of GNSS," in *Proceedings of the 29th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2016)*, Portland, Oregon, 2016.

[28] J. T. Curran and C. O'Driscoll, "Message Authentication as an Anti-Spoofing Mechanism," *Working Paper*, June 2017, dOI: 10.13140/RG.2.2.28625.12640 https://www.researchgate.net/publication/317950338_Message_Authentication_as_an_Anti-Spoofing_Mechanism.

[29] European Commission, "Commission Regulation (EC) No 2244/2003 as of 18 December 2003 laying down detailed provisions regarding satellite-based Vessel Monitoring Systems," 2003. [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX{%}3A32003R2244

[30] U. Kröner, H. Greidanus, R. Gallagher, M. Sironi, G. Azzalin, F. Littmann, P. Tebaldi, P. Timossi, and D. Shaw, "Report on Authentication in Fisheries Monitoring - Possible benefits of the Galileo GNSS system for the Vessel Monitoring System and fisheries control," Joint Research Center, Tech. Rep., 2009. [Online]. Available: http://cordis.europa.eu/publication/rcn/200910513{_}it.html

[31] European Commission, "Regulation (EC) No 561/2006 of the European Parliament and of the Council of 15 March 2006 on the harmonisation of certain social legislation relating to road transport and amending Council Regulations (EEC) No 3821/85 and (EC) No 2135/98 and repealing Council Regulation (EEC) No 3820/85," 2006. [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX{%}3A32006R0561

[32] ——, "Commission Implementing Regulation (EU) 2016/799 of 18 March 2016 implementing Regulation (EU) No 165/2014 of the European Parliament and of the Council laying down the requirements for the construction, testing, installation, operation and repair of tachographs and their components," 2016.

[33] E. G. Agency, "Development, Supply and Testing of a Galileo Open Service Authentication User Terminal (OS-NMA) for the GSA – Annex I to Invitation to Tender: Tender Specifications," 2017. [Online]. Available: https://www.gsa.europa.eu/development-supply-and-testing-galileo-open-service-authentication-user-terminal-os-nma-gsa

[34] I. Fernández-Hernández and G. Seco-Granados, "Galileo NMA Signal Unpredictability and Anti-Replay Protection," in *ICL-GNSS 2016*, 2016.

[35] European Union, "Galileo Open Service SIS ICD," 2016, http://ec.europa.eu/enterprise/policies/satnav/galileo/files/galileo_os_sis_icd_revised_3_en.pdf.

[36] J. T. Curran and C. O'Driscoll, "Message Authentication and Channel Coding," in *Proceedings of the 29th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2016)*, Portland, Oregon, 2016, pp. 2948 – 2959. [Online]. Available: https://www.ion.org/gnss/abstracts.cfm?paperID=4319

[37] D. M. Akos, "Who's Afraid of the Spoofer? GPS/GNSS Spoofing Detection via Automatic Gain Control (AGC)," *Navigation*, vol. 59, no. 4, pp. 281–290, dec 2012. [Online]. Available: http://doi.wiley.com/10.1002/navi.19

[38] A. Jafarnia Jahromi, "GNSS Signal Authenticity Verification in the Presence of Structural Interference," Ph.D. dissertation, University of Calgary, 2013. [Online]. Available: http://theses.ucalgary.ca/handle/11023/927

[39] GNSS-SDR, "An open source Global Navigation Satellite Systems software-defined receiver," http://gnss-sdr.org, [Accessed: 27 June 2017].

[40] Nuand, "bladeRF - the USB 3.0 Superspeed Software Defined Radio," https://www.nuand.com, [Accessed: 27 June 2017].

[41] L. Scott, "Anti-spoofing & authenticated signal architectures for civil navigation systems," *Proceedings of the Institute of Navigation GPS/GNSS 2003 conference*, pp.

1543–1552, 2003. [Online]. Available: http://www.ion.org/search/view{_}abstract.cfm?jp=p{&}idno=5339

[42] M. G. Kuhn, "An asymmetric security mechanism for navigation signals," in *International Workshop on Information Hiding, IH*, ser. Lecture Notes in Computer Science, J. Fridrich, Ed., vol. 3200. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 239–252. [Online]. Available: http://www.mendeley.com/catalog/asymmetric-security-mechanism-navigation-signals/http://link.springer.com/book/10.1007/b104759

[43] A. J. Jahromi, A. Broumandan, S. Daneshmand, G. Lachapelle, and R. T. Ioannides, "Galileo signal authenticity verification using signal quality monitoring methods," in *2016 International Conference on Localization and GNSS (ICL-GNSS)*. IEEE, jun 2016, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/document/7533684/

[44] D. Borio, F. Dovis, H. Kuusniemi, and L. Lo Presti, "Impact and detection of gnss jammers on consumer grade satellite navigation receivers," *Proceedings of the IEEE*, vol. 104, no. 6, pp. 1233–1245, June 2016.

[45] D. Borio, C. O'Driscoll, and J. Fortuny, "GNSS jammers: Effects and countermeasures," *6th ESA Workshop on Satellite Navigation Technologies: Multi-GNSS Navigation Technologies Galileo's Here, NAVITEC 2012 and European Workshop on GNSS Signals and Signal Processing*, 2012.

[46] A. Rügamer and D. Kowalewski, "Jamming and Spoofing of GNSS Signals – An Underestimated Risk?!" *FIG Working Week - From the Wisdom of the Ages to the Challenges of the Modern World, 2015*, no. May 2015, pp. 1–21, 2015.

[47] D. Borio and P. Closas, "A Fresh Look at GNSS Anti-Jamming," *Inside GNSS*, pp. 54–61, 2017.

[48] G. Caparra, S. Ceccato, F. Formaggio, N. Laurenti, and S. Tomasin, "Low power selective denial of service attacks against gnss," 09 2018.

[49] A. J. Van Dierendonck, "GPS Receivers," in *Global Positioning System: Theory and Applications*, B. W. Parkinson and J. J. J. Spilker, Eds., 1996.

[50] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, ser. Discrete Mathematics and Its Applications. Taylor & Francis, 1996. [Online]. Available: http://books.google.it/books?id=nSzoG72E93MC

[51] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz, "Extensible Authentication Protocol (EAP)," Internet Requests for Comments, RFC Editor, RFC 3748, June 2004. [Online]. Available: https://tools.ietf.org/pdf/rfc3748.pdf

[52] J. Arkko and H. Haverinen, "Extensible Authentication Protocol Method for 3rd Generation, Authentication and Key Agreement (EAP-AKA)," Internet Requests for Comments, RFC Editor, RFC 4187, January 2006. [Online]. Available: https://tools.ietf.org/html/rfc4187

[53] S. Alt, P.-A. Fouque, G. Macario-rat, C. Onete, and B. Richard, *A Cryptographic Analysis of UMTS/LTE AKA*. Cham: Springer International Publishing, 2016, pp. 18–35. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-39555-5_2

[54] ETSI, "Universal Mobile Telecommunications System (UMTS); LTE; 3G security; Security architecture," vol. 3GPP TS 33.102 - V8.6.0 - Release 8, pp. 0–69, 2010.

[55] J. Cichonski and J. Franklin, "LTE Security – How Good Is It?" presented at the RSA Conference in April 2015 in San Francisco, USA. [Online]. Available: https://www.rsaconference.com/writable/presentations/file_upload/tech-r03_lte-security-how-good-is-it.pdf

[56] Netmanias, "LTE Security I - Security Concept and Authentication," pp. 1–12, 2013. [Online]. Available: http://www.slideshare.net/Netmanias/lte-security-isecurity-concept-and-authentication-en

[57] H. Tian, L. Pang, and Y. Wang, "Key management protocol of the IEEE 802.16e," *Wuhan University Journal of Natural Sciences*, vol. 12, no. 1, pp. 59–62, 2007.

[58] D. Akin, "802.11i Authentication and Key Management (AKM)," Certified Wireless Network Professional (CWNP), Tech. Rep. May, 2005.

[59] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol, Version 1.2," Internet Requests for Comments, RFC Editor, RFC 5246, August 2008. [Online]. Available: https://tools.ietf.org/html/rfc5246

[60] Fu-Kuan Tu, Chi-Sung Laih, and Hsu-Hung Tung, "On key distribution management for conditional access system on pay-TV system," *IEEE Transactions on Consumer Electronics*, vol. 45, no. 1, pp. 151–158, 1999. [Online]. Available: http://ieeexplore.ieee.org/document/754430/

[61] G. Caparra, S. Ceccato, S. Sturaro, and N. Laurenti, "A key management architecture for gnss open service navigation message authentication," in *2017 European Navigation Conference (ENC)*, May 2017, pp. 287–297.

[62] S. Ceccato, "A key management scheme for access control to gnss services," Master's thesis, University of Padova, 2016.

[63] G. Caparra, S. Sturaro, and N. Laurenti, "Novel and Advanced Authentication Techniques for GNSS - A GOSSIP, A PLAY Technical Note 3," 2016.

[64] A. A. Yavuz, "ETA: efficient and tiny and authentication for heterogeneous wireless systems," in *ACM conference on Wireless network security, WiSec*, 2013, pp. 67–72. [Online]. Available: http://dl.acm.org/citation.cfm?id=2462108

[65] A. Petzoldt, S. Bulygin, and J. Buchmann, "Selecting Parameters for the Rainbow Signature Scheme," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010, vol. 6061 LNCS, pp. 218–240. [Online]. Available: http://link.springer.com/10.1007/978-3-642-12929-2{_}16

[66] N. P. Smart, V. Rijmen, B. Gierlichs, K. G. Paterson, M. Stam, B. Warinschi, and G. Watson, "Algorithms, Key Size and Parameters Report," ENISA, Tech. Rep., 2014. [Online]. Available: https://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014

[67] GMV, "Galileo Commercial Service (CS)," http://www.navipedia.net/index.php/Galileo_Commercial_Service_(CS), [Accessed: 21 May 2018].

[68] A. Fiat and M. Naor, "Advances in Cryptology — CRYPTO' 93," *CRYPTO '93: Proceedings of the 13th Annual International Conference on Advances in Cryptology*, vol. 773, pp. 480–491, 1994. [Online]. Available: http://link.springer.com/10.1007/3-540-48329-2

[69] B. Briscoe, "MARKS: Zero side effect multicast key management using arbitrarily revealed key sequences," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1736, pp. 301–320, 1999.

[70] L. Veltri, S. Cirani, S. Busanelli, and G. Ferrari, "A novel batch-based group key management protocol applied to the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2724–2737, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2013.05.009

[71] M. Naor and B. Pinkas, "Efficient trace and revoke schemes," *International Journal of Information Security*, vol. 9, no. 6, pp. 411–424, 2010.

[72] M. Abdalla, Y. Shavitt, and A. Wool, "Key management for restricted multicast using broadcast encryption," *{IEEE/ACM} Transactions on Networking*, vol. 8, no. 4, pp. 443–454, 2000.

[73] J. Fan and M. H. Ammar, "HySOR : Group Key Management with Collusion-Scalability Tradeoffs Using a Hybrid Structuring of Receivers," vol. 00, no. 1, pp. 196–201, 2002.

[74] J. Liu and C. Wang, "Exclusive Key Based Group Rekeying Protocols," no. 61173189, pp. 1–29.

[75] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16–30, Feb 2000.

[76] D. Wallner, E. Harder, and R. Agee, "Key management for multicast: Issues and architectures," United States, Tech. Rep., 1999.

[77] K. Y. Chou, Y. R. Chen, and W. G. Tzeng, "An efficient and secure group key management scheme supporting frequent key updates on Pay-TV systems," *APNOMS 2011 - 13th Asia-Pacific Network Operations and Management Symposium: Managing Clouds, Smart Networks and Services, Final Program*, 2011.

[78] Jia-Yin Tian, Cheng Yang, Yi-chun Zhang, and Jian-Bo Liu, "Distributed Key Management Scheme for Large Scale Pay-TV Application," pp. 30–33, 2010.

[Online]. Available: http://ieeexplore.ieee.org/ielx5/5453385/5453559/05453655.pdf?tp={&}arnumber=5453655{&}isnumber=5453559

[79] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology – EUROCRYPT 2005*, R. Cramer, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 440–456.

[80] V. Shoup, "Lower bounds for discrete logarithms and related problems," in *Advances in Cryptology — EUROCRYPT '97*, W. Fumy, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 256–266.

[81] D. R. L. Brown, "Generic groups, collision resistance, and ecdsa," *Designs, Codes and Cryptography*, vol. 35, no. 1, pp. 119–152, Apr 2005. [Online]. Available: https://doi.org/10.1007/s10623-003-6154-z

[82] A. W. Dent, "Adapting the weaknesses of the random oracle model to the generic group model," in *Advances in Cryptology — ASIACRYPT 2002*, Y. Zheng, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 100–109.

[83] C. P. Schnorr and M. Jakobsson, "Security of signed elgamal encryption," in *Advances in Cryptology — ASIACRYPT 2000*, T. Okamoto, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 73–89.

[84] N. Koblitz and A. Menezes, "Another look at generic groups," in *Advances in Mathematics of Communications*, 2006, pp. 13–28.

[85] L. Parolini, "Security evaluation of a key management scheme based on bilinear maps on elliptic curves," 2019. [Online]. Available: http://tesi.cab.unipd.it/62451/

[86] A. J. Menezes, T. Okamoto, and S. A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *IEEE Transactions on Information Theory*, vol. 39, no. 5, pp. 1639–1646, Sep. 1993.

[87] J. Silverman, *The Arithmetic of Elliptic Curves*, ser. Graduate Texts in Mathematics. Springer New York, 2009. [Online]. Available: https://books.google.com.au/books?id=Z90CA_EUCCkC

[88] G. Adj, A. Menezes, T. Oliveira, and F. Rodríguez-Henríquez, "Computing discrete logarithms in $\mathbb{F}_{3^{(6\cdot 137)}}$ and $\mathbb{F}_{3^{(6\cdot 163)}}$ using magma," in *Arithmetic of Finite Fields*, Ç. K. Koç, S. Mesnager, and E. Savaş, Eds. Cham: Springer International Publishing, 2015, pp. 3–22.

[89] R. Barbulescu, P. Gaudry, A. Guillevic, and F. Morain, "Improving nfs for the discrete logarithm problem in non-prime finite fields," in *Advances in Cryptology – EUROCRYPT 2015*, E. Oswald and M. Fischlin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 129–155.

[90] R. Granger, T. Kleinjung, and J. Zumbrägel, "Breaking '128-bit secure' supersingular binary curves," in *Advances in Cryptology – CRYPTO 2014*, J. A. Garay and R. Gennaro, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 126–145.

[91] T. Teruya, K. Saito, N. Kanayama, Y. Kawahara, T. Kobayashi, and E. Okamoto, "Constructing symmetric pairings over supersingular elliptic curves with embedding degree three," in *Pairing-Based Cryptography – Pairing 2013*, Z. Cao and F. Zhang, Eds. Cham: Springer International Publishing, 2014, pp. 97–112.

[92] F. Hess, N. P. Smart, and F. Vercauteren, "The eta pairing revisited," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4595–4602, Oct 2006.

[93] P. Gaudry, F. Hess, and N. P. Smart, "Constructive and destructive facets of weil descent on elliptic curves," *Journal of Cryptology*, vol. 15, no. 1, pp. 19–46, Mar 2002. [Online]. Available: https://doi.org/10.1007/s00145-001-0011-x

[94] I. Fernández-Hernández, J. Simón, R. Blasi, C. Payne, T. Miquel, and J. P. Boyero, "The Galileo Commercial Service: Current Status and Prospects," in *European Navigation Conference, (ENC-GNSS)*, Rotterdam, 2014. [Online]. Available: http://gpsworld.com/enc-gnss-2014-program-now-online/http://mycoordinates.org/the-galileo-commercial-service-current-status-and-prospects/

[95] B. Lynn, "Ben Lynn June 2007," no. June, pp. 1–126, 2007.

[96] R. Blank, A. Secretary, P. D. Gallagher, E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, "Nist special publication 800-57, recommendation for key management part 1: General (revision 3)," 2012.

[97] Lucia Angeloni, "Un Sistema ad Alta Precisione," *TIR - #Trasporti #Innovazione #Rete*, vol. 212, no. April, pp. 16–19, 2018.

[98] J. A. Garcia, M. Navarro, M. Cordero, and J. Miguez, "E6 CS Encryption Flexibility-User Receiver Impact Based on Field Testing," pp. 1–32, 2016.

[99] F. Gerlin, N. Laurenti, G. Naletto, G. Vallone, P. Villoresi, L. Bonino, S. Mottini, and Z. Sodnik, "Design optimization for quantum communications in a gnss intersatellite network," in *2013 International Conference on Localization and GNSS (ICL-GNSS)*, June 2013, pp. 1–6.

[100] H. Hemmati, *Near Earth Laser Communications*. CRC Press, 2009, ISBN 978-0824753818.

[101] F. Ardizzon, "Determinazione del secret key rate non asintotico per una rete qkd inter-satellitare con secret sharing," Master's thesis, University of Padova, 2019.

[102] C. E. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, Oct 1949.

[103] "Spoofing Incident Report: An Illustration of Cascading Security Failure," http://www.insidegnss.com, Inside GNSS news, [Accessed: 15 January 2018].

[104] P. Montgomery, T. Humphreys, and B. Ledvina, "Receiver-autonomous spoofing detection: Experimental results of a multi-antenna receiver defense against a portable civil gps spoofer," *Proceedings of the Institute of Navigation, National Technical Meeting*, vol. 1, pp. 124–130, 01 2009.

[105] K. Wesson, D. Shepard, J. Bhatti, and T. Humphreys, "An evaluation of the vestigial signal defense for civil gps anti-spoofing," vol. 4, 01 2011.

[106] S. Ceccato, F. Formaggio, G. Caparra, N. Laurenti, and S. Tomasin, "Exploiting side-information for resilient gnss positioning in mobile phones," in *2018 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, April 2018, pp. 1515–1524.

[107] H. Kuusniemi, E. Airos, M. Z. H. Bhuiyan, and T. Kröger, "Gnss jammers: how vulnerable are consumer grade satellite navigation receivers?" *European Journal of Navigation*, vol. 10, pp. 14–21, 08 2012.

[108] L. Huang and Q. Yang, "Low-cost gps simulator," DEFCON 23, 2015.

[109] Septentrio, "Gps spoofing: is your receiver ready for an attack?" http://www.septentrio.com/insights/gps-spoofing-your-receiver-ready-attack, 2017.

[110] T. Frost and G. Buesnel, "Spoofing gnss timing receivers," ITSF presentation, 2016.

[111] J. Nielsen, V. Dehghanian, and G. Lachapelle, "Effectiveness of GNSS Spoofing Countermeasure Based on Receiver CNR Measurements," *International Journal of Navigation and Observation*, vol. 2012, pp. 1–9, 2012. [Online]. Available: http://www.hindawi.com/journals/ijno/2012/501679/

[112] J. T. Curran and A. Broumendan, "On the use of Low-Cost IMUs for GNSS Spoofing Detection in Vehicular Applications," in *International Technical Symposium on Navigation and Timing (ITSNT) 2017*, Toulouse, France, 2017.

[113] A. Broumandan, S. Daneshmand, A. Jafarnia-jahromi, A. Broumandan, and G. Lachapelle, "A low-complexity GPS anti-spoofing method using a multi-antenna array A Low-Complexity GPS Anti-Spoofing Method Using a Multi-Antenna Array," in *ION GNSS 2012*, no. February, 2012.

[114] R. S. Campos, "Evolution of positioning techniques in cellular networks, from 2G to 4G," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.

[115] N. O. Tippenhauer, K. B. Rasmussen, C. Pöpper, and S. Čapkun, "Attacks on public wlan-based positioning systems," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '09. New York, NY, USA: ACM, 2009, pp. 29–40. [Online]. Available: http://doi.acm.org/10.1145/1555816.1555820

[116] F. V. Diggelen, *Assisted GPS, GNSS, and SBAS*. Artech House, 2009, ISBN 978-1-59693-374-3.

[117] M. Monnerat, "Agnss standardization. the path to success in location-based services." *GNSS Magazine*, pp. 22–23, 2008.

[118] "3GPP TS 44.031," 3rd Generation Partnership Project; Technical Specifcation Group GSM/ EDGE Radio Access Network; Location Services (LCS); Mobile Station (MS)—Serving Mobile Location Centre (SMLC) Radio Resource LCS Protocol (RRLP), Standard, 2010.

[119] "3GPP TS 144 118," 3rd Generation Partnership Project; Digital cellular telecommunications system (Phase 2+) (GSM); Mobile radio interface layer 3 specification; Radio Resource Control (RRC) protocol; Technical Specifcation Group GSM, Standard, 2017.

[120] "Secure User Plane Location Architecture," Open Mobile Alliance, Standard, 2008.

[121] W. Kang and Y. Han, "SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2906–2916, 2015.

[122] Z. A. Deng, Y. Hu, J. Yu, and Z. Na, "Extended Kalman filter for real time indoor localization by fusing WiFi and smartphone inertial sensors," *Micromachines*, vol. 6, no. 4, pp. 523–543, 2015.

[123] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. C. Soh, and L. Xie, "Fusion of WiFi, smartphone sensors and landmarks using the kalman filter for indoor localization," *Sensors (Switzerland)*, vol. 15, no. 1, pp. 715–732, 2015.

[124] A. Rodriguez and U. Shala, "Indoor Positioning using Sensor-fusion in Android Devices," Ph.D. dissertation, Kristianstad University, 2011.

[125] I. C. W. Group, "NAVSTAR GPS Space Segment/Navigation User Segment Interface," in *IS-GPS-200H*, 2015, pp. 1–224. [Online]. Available: http://www.ion.org/gnss/abstracts.cfm?paperID=862

[126] C. Limited, "GPS Test," https://play.google.com/store/apps/details?id=com.chartcross.gpstest&hl=it.

[127] "Keep your device unlocked when it's at a trusted place," https://support.google.com/nexus/answer/6093922?hl=en.

[128] Starlino, "A Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications," http://www.starlino.com/imu_guide.html, [Posted: 29 December 2009].

[129] C. Tanil, S. Khanafseh, and B. Pervan, "Impact of wind gusts on detectability of gps spoofing attacks using raim with ins coupling," 04 2015.

[130] ——, "An ins monitor against gnss spoofing attacks during gbas and sbas-assisted aircraft landing approaches," 09 2016.

[131] GSA, "Using gnss raw measurements on android devices," 01 2018.

[132] G. Falco, M. Pini, and G. Marucco, "Loose and tight gnss/ins integrations: Comparison of performance assessed in real urban scenario," *Sensors*, Jan 2017.

[133] G. Falco, G. Einicke, J. Malos, and F. Dovis, "Performance analysis of constrained loosely coupled gps/ins integration solutions," *Sensors (Basel, Switzerland)*, vol. 12, pp. 15 983–6007, 12 2012.

[134] V. Gikas and H. Perakis, "Rigorous performance evaluation of smartphone gnss/imu sensors for its applications," *Sensors*, Aug 2016.

[135] S. Lo, Y. H. Chen, T. Reid, A. Perkins, T. Walter, and P. Enge, "Keynote: The benefits of low cost accelerometers for gnss anti-spoofing," in *ION 2017 Pacific PNT Meeting*, 2016.

[136] B. A. and L. G., "Spoofing detection using gnss/ins/odometer coupling for vehicular navigation," *Sensors*, Apr 2018.

[137] M. Kok, J. D. Hol, and T. B. Schön, "Using Inertial Sensors for Position and Orientation Estimation," vol. 11, no. 1-2, 2017, pp. 1–153.

[138] A. Quinchia, G. Falco, E. Falletti, F. Dovis, and C. Ferrer, "A comparison between different error modeling of mems applied to gps/ins integrated systems," *Sensors (Basel, Switzerland)*, vol. 13, pp. 9549–88, 08 2013.

[139] X. Niu, Q. Chen, Q. Zhang, H. Zhang, J. Niu, K. Chen, C. Shi, and J. Liu, "Using Allan variance to analyze the error characteristics of GNSS positioning," *GPS solutions*, vol. 18, no. 2, pp. 231–242, 2014.

[140] J. Rankin, "GPS and differential gps: an error model for sensor simulation," in *Position Location and Navigation Symposium*, 1994, pp. 260–260.

[141] S. M. Kay, *Fundamentals of statistical signal processing, Vol. II: Detection Theory*, 1998.

[142] M. S. Mark Pedley and Z. Baranski, "Freescale Sensor Fusion Kalman Filter," available at https://github.com/memsindustrygroup/Open-Source-Sensor-Fusion/tree/master/docs.

[143] Y. Liu, S. Li, Q. Fu, and Z. Liu, "Impact assessment of gnss spoofing attacks on ins/gnss integrated navigation system," in *Sensors*, 2018.

[144] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, "Unmanned aircraft capture and control via gps spoofing," *J. Field Robotics*, vol. 31, pp. 617–636, 2014.

[145] 3GPP, "NR and NG-RAN Overall Description - Rel. 15," TS 38.300, 2018.

[146] A. Ranganathan, N. O. Tippenhauer, B. Škorić, D. Singelée, and S. Čapkun, "Design and implementation of a terrorist fraud resilient distance bounding system," in *Computer Security – ESORICS 2012*, S. Foresti, M. Yung, and F. Martinelli, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 415–432.

[147] M. Kuhn, H. Luecken, and N. O. Tippenhauer, "Uwb impulse radio based distance bounding," in *2010 7th Workshop on Positioning, Navigation and Communication*, March 2010, pp. 28–37.

[148] A. Mitrokotsa, C. Onete, and S. Vaudenay, "Mafia fraud attack against the rČ distance-bounding protocol," in *2012 IEEE International Conference on RFID-Technologies and Applications (RFID-TA)*, Nov 2012, pp. 74–79.

[149] M. Fischlin and C. Onete, "Terrorism in distance bounding: Modeling terrorist-fraud resistance," in *Applied Cryptography and Network Security*, M. Jacobson, M. Locasto, P. Mohassel, and R. Safavi-Naini, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 414–431.

[150] S. Brands and D. Chaum, "Distance-bounding protocols," in *Advances in Cryptology — EUROCRYPT '93*, T. Helleseth, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 344–359.

[151] S. Capkun and J. Hubaux, "Secure positioning in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 221–232, Feb 2006.

[152] N. O. Tippenhauer and S. Čapkun, "Id-based secure distance bounding and localization," in *Proceedings of the 14th European Conference on Research in Computer Security*, ser. ESORICS'09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 621–636. [Online]. Available: http://dl.acm.org/citation.cfm?id=1813084.1813135

[153] S. Sun, T. S. Rappaport, R. W. Heath, A. Nix, and S. Rangan, "MIMO for millimeter-wave wireless communications: beamforming, spatial multiplexing, or both?" *IEEE Communications Magazine*, vol. 52, no. 12, pp. 110–121, December 2014.

[154] J. Chiang, J. Haas, and Y.-C. Hu, "Secure and precise location verification using distance bounding and simultaneous multilateration," 01 2009, pp. 181–192.