



**Politecnico
di Torino**

Politecnico di Torino

Cybersecurity Engineering

A.a. 2025/2026

Graduation Session October 2026

Resilient PNT for Maritime Applications: A Multi-Layered Architecture to mitigate GNSS Vulnerabilities

Supervisors:

Andrea Nardin
Andrea Bellés Ferreres

Candidate:

Marco Sabbatini

Ringraziamenti

Rivendico con orgoglio l'impegno, la disciplina e le rinunce che mi hanno reso il principale artefice di questo traguardo: è il risultato diretto della mia determinazione e dei sacrifici con cui ho affrontato ogni bivio e ogni ostacolo. È stato un viaggio intenso, veloce e a tratti difficile. Ma se la fatica è stata tutta mia, la verità è che da solo non ce l'avrei mai fatta: la forza di non perdermi d'animo e di arrivare fino in fondo l'ho trovata anche nell'affetto e nella presenza di chi ha sempre avuto fiducia in me.

Questa tesi, ultima tappa del cammino, deve molto al mio relatore e alla mia correlatrice. Al Prof. Andrea Nardin va la mia riconoscenza per la professionalità e per l'opportunità formativa concessami nell'affrontare questo progetto. Alla Dr. Andrea Bellés Ferreres il mio grazie più sentito per le energie e per la pazienza con cui ha saputo consigliarmi, sopportarmi e comprendere i momenti più complessi del percorso.

Ma un traguardo non si esaurisce nelle sue ultime pagine: dietro c'è stato qualche anno di strada, e non l'ho mai percorsa da solo.

Alla mia famiglia, che ha reso possibile tutto questo ben prima che io sapessi dove mi avrebbe portato: grazie per non avermi mai fatto mancare nulla, per il sostegno continuo e per avermi donato la serenità, i mezzi e la libertà di scegliere sempre la mia strada, anche quando quella strada mi portava lontano da casa. Ogni risultato che ho raggiunto porta con sé qualcosa di voi.

Ai miei nonni, che pur non sapendo molto di quello che faccio mi hanno sempre sostenuto senza riserve, con la semplicità di chi ama senza bisogno di capire. Grazie per ogni attesa paziente, per ogni ritorno accolto come una festa. Questo traguardo è anche vostro.

A Benedetta e alla sua famiglia, per il calore e la vicinanza che non mi hanno mai fatto mancare. A te, Benedetta, qualsiasi parola risulterebbe riduttiva, incapace di racchiudere davvero ciò che hai rappresentato per me in questo percorso. Sei stata una sorgente continua di ottimismo e di coraggio, la presenza che, nonostante la distanza, ha saputo starmi accanto giorno dopo giorno, accorciando ogni chilometro fino a farlo scomparire. Grazie per avermi spronato a diventare una persona migliore, per aver sopportato con infinita pazienza le mie ore interminabili davanti a uno

schermo in questi due anni, e per avermi riportato, ogni volta che ce n'era bisogno, alla vita che mi aspettava oltre la scrivania. Senza di te questo traguardo non avrebbe avuto lo stesso valore; spero, con tutto il cuore, di averti resa orgogliosa.

Agli amici che Torino mi ha fatto incontrare, compagni di studio e di vita lungo tutti questi anni universitari: grazie per i progetti affrontati fianco a fianco, per le notti trascorse a inseguire una scadenza, per le passioni condivise e coltivate insieme. Avete trasformato una città lontana in un luogo familiare, e questo capitolo in un ricordo che porterò con me a lungo.

Agli amici di sempre, da cui la vita e le strade intraprese mi hanno portato lontano: grazie perché a ogni ritorno basta un istante per annullare i mesi trascorsi e farmi sentire di nuovo a casa, ritrovando intatta una storia che il tempo non riesce a consumare. Tra loro, una dedica fraterna va a Gabriele: una spalla autentica su cui so di poter contare a occhi chiusi, oggi come sempre.

A ciascuno di voi va la mia gratitudine più profonda: questo traguardo porta la mia firma, ma il suo valore appartiene anche a voi, a chi c'è stato e a chi, in un modo o nell'altro, ha camminato accanto a me.

Table of Contents

Ringraziamenti	III
List of Tables	IX
List of Figures	X
Acronyms	XIII
1 Introduction	1
2 GNSS Fundamentals	5
2.1 Introduction	5
2.2 GNSS Constellations	5
2.3 GNSS Segments	6
2.4 Structure of a GNSS Signal	6
2.5 The Galileo E1 Signal	6
2.6 Receiver Signal Processing	7
2.6.1 Front-End and Signal Acquisition	7
2.6.2 Signal Tracking and Quality Monitoring	7
2.6.3 Observables and Navigation Data	8
2.7 Single Point Positioning	9
2.7.1 Weighted Least Squares Estimation	10
3 GNSS Security	13
3.1 Introduction	13
3.2 GNSS Threats	13
3.3 Detection and Mitigation of GNSS Interference	14
3.4 Anti-Spoofing and Anti-Jamming Techniques	15
3.4.1 Non-Cryptographic Detection Parameters	16
3.4.2 Cryptographic Authentication and Encryption	17
3.4.3 Open Service Navigation Message Authentication	18

3.4.4	Signal Authentication Service	18
3.4.5	Chips-Message Robust Authentication	18
3.5	Summary	19
4	Galileo OSNMA: Principles, Architecture, and Implementation	21
4.1	Introduction	21
4.2	OSNMA Data Structures	21
4.2.1	OSNMA Field Structure	22
4.2.2	The HKROOT Message	22
4.2.3	The MACK Message	24
4.3	OSNMA Receiver Workflow	25
4.3.1	Receiver Requirements	25
4.3.2	OSNMA Data Retrieval	27
4.3.3	Chain of Trust and Start Modes	29
4.3.4	Processing of the Authentication Material	30
4.4	Software Architecture and Implementation	34
4.4.1	CSSRlib for Galileo-Only SPP	34
4.4.2	OSNMAlib for OSNMA Processing	35
4.4.3	Processing Pipeline	36
4.4.4	Diagnostic Outputs	37
5	OSNMA Behaviour in Real Reception Scenarios	41
5.1	Chapter Overview	41
5.2	Experimental Setup	41
5.3	Aggregated View of the Authentication Process	43
5.3.1	Nominal Scenario	43
5.3.2	Constrained Scenario	43
5.3.3	Signal Conditions Behind the Difference	44
5.4	The Full Diagnostic Output	45
5.5	Single-Satellite Events	46
5.5.1	Initialisation and the TTFAF Boundary	49
5.5.2	TESLA Delay and the Cross-Authenticator Role	50
5.5.3	Satellites Never Cross-Authenticated	51
5.5.4	The Memory Mechanism Under Signal Degradation	52
5.5.5	Abrupt Zeroing of the OSNMA Field	53
5.5.6	Progressive Loss of Authentication	54
5.6	Impact of Authentication on the Position Solution	55
5.6.1	Nominal Scenario	55
5.6.2	Constrained Scenario	56
5.6.3	Retrospective Selection	57
5.7	Summary	58

6	Live Authentication and Start-Mode Analysis	59
6.1	Chapter Overview	59
6.2	Start Modes and Time to First Authenticated Fix	59
6.2.1	Nominal Setup	59
6.2.2	Cold and Warm Starts having Same TTFAF	60
6.2.3	Warm and Hot Starts Collapsing, and a Proposed Change	60
6.3	Constrained Scenario with the ZED-X20P	61
6.4	SDR-Based Spoofing Simulation	61
6.5	A Jammertest Recording With Declared Meaconing and Spoofing	62
6.5.1	Aggregated View	63
6.5.2	Skyplot and Reception Conditions	65
6.5.3	Full Diagnostic Timeline	66
6.5.4	Comparison With the Other Recordings	67
7	GNSS Interference Detection and Classification	69
7.1	Introduction	69
7.2	Machine Learning for GNSS Threats	70
7.3	Design Constraints	71
7.4	Interference Signatures	71
7.5	Anomaly Detection	72
7.6	Attack Classification	73
8	Experimental Setup and Scenarios	75
8.1	Objectives	75
8.2	Jammertest 2025	75
8.3	Dataset	76
8.3.1	Extraction	76
8.3.2	Labels	76
8.3.3	Features and Preprocessing	77
8.4	Models	78
8.4.1	Detector	78
8.4.2	Classifier	78
8.4.3	Baselines	80
8.5	Evaluation	82
8.5.1	Detector Protocol	82
8.5.2	Classifier Protocol	82
8.5.3	Metrics	83
9	Results and Discussion	85
9.1	Anomaly Detection	85
9.1.1	Threshold Transfer	85

9.1.2	Block 0	86
9.2	Attack Classification	86
9.2.1	Jamming Failure Case	87
9.3	Temporal vs Memoryless Models	88
9.3.1	Detection	88
9.3.2	Classification	89
10	Conclusions and Future Work	91
10.1	Summary	91
10.2	Main Findings	91
10.3	Limitations	92
10.4	Future Work	93
A	OSNMAlib output README	95
	Bibliography	105

List of Tables

4.1	NMAS values and receiver interpretation.	23
5.1	Reference recordings used in this chapter.	42
6.1	Average TTFAF over the eight rooftop recordings, by start mode. . .	60
6.2	Reference recording used in this section.	63
6.3	Recordings used to characterise OSNMA behaviour across the thesis, by attack type and how ground truth was established.	67
7.1	The two models side by side.	69
7.2	Expected joint signatures.	72
8.1	Class composition after re-labelling.	76
8.2	Nominal reference state.	77
8.3	Retained features and their role.	78
8.4	Preprocessing steps and their motivation.	79
8.5	Configuration of the autoencoder.	79
8.6	One-Class SVM configuration.	81
8.7	Random Forest baseline configuration.	81
9.1	LSTM detector, leave-one-block-out, denial epochs excluded.	85
9.2	Spread induced by the choice of validation block.	86
9.3	Classifier, all test windows.	86
9.4	Classifier, excluding all-denial windows.	87
9.5	Measured effect on E1 of two 47 dBm CW scenarios.	87
9.6	One-Class SVM, leave-one-block-out, denial epochs excluded.	88
9.7	Detector against One-Class SVM, denial epochs excluded.	88
9.8	Random Forest, five draws, test excluding all-denial epochs. PR- AUC is given with the class share of that draw in brackets.	89
9.9	Classifier against Random Forest, means over the five draws, test excluding all-denial units.	89

List of Figures

4.1	Structure of the Galileo E1-B I/NAV page and sub-frame, showing the position of the 40-bit OSNMA field within each page[39].	22
4.2	Structure of the Merkle tree used by OSNMA[39].	24
4.3	Overall OSNMA structure. Author's elaboration synthesized from the data structures defined in [39].	26
4.4	High-level OSNMA chain of trust. Each element can only be used once the previous one has been verified.	29
4.5	Processing of the OSNMA authentication material.	30
4.6	Recursive verification of a TESLA chain key against a previously authenticated key [5].	33
4.7	Conceptual relation between the received tag, the locally stored navigation data and the verified TESLA key used for tag recomputation[45].	34
5.1	Nominal static setup: rooftop antenna and its position on the site. .	42
5.2	Nominal scenario: tracked, complete, authenticated and missed authentication counts, one point per 30-second subframe.	44
5.3	Constrained scenario: the same four counts. The non-zero red peaks mark satellites that had complete navigation data in the previous subframe and did not reach an authenticated state in the current one.	44
5.4	Constrained scenario: skyplot of the Galileo satellites reported by the receiver, built from UBX NAV-SAT.	45
5.5	Constrained scenario: C/N_0 timeline per satellite, showing the north/south asymmetry produced by the building.	45
5.6	Nominal scenario, full diagnostic output, shown for comparison with Figure 5.7.	47
5.7	Constrained scenario, full diagnostic output: extraction state and authentication timelines on top, C/N_0 , elevation, azimuth and PDOP below, all on a common time axis.	48

5.8	Initialisation and the TTFAF boundary: extraction state (top) and authentication timeline (bottom) around the first authenticated subframe.	49
5.9	TESLA delay and the cross-authenticator role. Red: self-authentication with no tags extracted in the current subframe. Yellow: cross-authentication of a satellite broadcasting no OSNMA bits.	50
5.10	A satellite never cross-authenticated: scattered page extraction, an empty authentication row, and the C/N_0 trace explaining both. . .	51
5.11	The memory mechanism: page losses in the extraction timeline with an uninterrupted cross-authenticated state below.	52
5.12	Abrupt zeroing of the OSNMA field: the violet outline disappears while extraction and signal quality remain nominal.	53
5.13	Progressive loss of authentication on SV 10, with the incomplete-extraction states marked in black and the unverified stretch marked in red.	54
5.14	Nominal scenario: all_galileo (top) against auth_only (bottom) SPP ENU error.	55
5.15	Constrained scenario: all_galileo (top) against auth_only (bottom) SPP ENU error. The two traces no longer overlap.	56
5.16	Constrained scenario: the step in the auth_only error (left) produced when SV 10 completes its authentication, absent from the all_galileo error (right).	57
6.1	TTFAF per recording and start mode over the eight rooftop sessions.	60
6.2	Constrained scenario with the ZED-X20P: aggregated timeline of tracked, complete and authenticated satellites.	62
6.3	SDR spoofing setup: GNU Radio, rubidium-disciplined SDR and receiver.	63
6.4	Spoofed circular trajectory visualized in u-center 2.	64
6.5	C/N_0 timeline of the SDR spoofing scenario.	64
6.6	Number of satellites tracked by the receiver in that epoch.	65
6.7	Number of satellites used in position estimation in that epoch. . . .	65
6.8	Tracked, complete, authenticated and missed-authentication counts over the 17:30–18:00 period. The nominal segment (17:30–17:45) matches the clear-sky case of Chapter 5; the declared attack begins at 17:45 and ends in a sustained loss of authentication that persists to the end of the recording.	65
6.9	Skyplot of the Galileo satellites tracked during the recording. . . .	66
6.10	Full diagnostic output for the 17:30–18:00 recording: satellite extraction state, navigation-data authentication timeline and C/N_0 , 2s bins.	67

Acronyms

ADC Analog-to-Digital Converter

ADKD Authentication Data and Key Delay

AGC Automatic Gain Control

CDMA Code Division Multiple Access

CHIMERA Chips-Message Robust Authentication

CID Chain Identifier

COP Cut-Off Point

CPKS Chain and Public Key Status

CSV Comma-Separated Values

CW Continuous Wave

DLR Deutsches Zentrum für Luft- und Raumfahrt

DSM Digital Signature Message

DVB-T Digital Video Broadcasting – Terrestrial

ENU East-North-Up

FDE Fault Detection and Exclusion

GNSS Global Navigation Satellite System

GPS Global Positioning System

GSC European GNSS Service Centre

GST Galileo System Time

HKROOT Header and Root Key

ID Identifier

IMU Inertial Measurement Unit

ITN Intermediate Tree Nodes

JSON JavaScript Object Notation

KROOT TESLA Root Key

LHCP Left-Hand Circular Polarization

LOS Line of Sight

LSTM Long Short-Term Memory

MAC Message Authentication Code

MACK MAC and Key

MACSEQ MAC Sequence

MAD Median Absolute Deviation

MEO Medium Earth Orbit

MID Merkle Tree Node Identifier

MPD Multi-Peak Detection

MSE Mean Squared Error

NMA Navigation Message Authentication

NMAS NMA Status

NME Navigation Message Encryption

OC-SVM One-Class Support Vector Machine

OSNMA Open Service Navigation Message Authentication

PDOP Position Dilution of Precision

PNT Position Navigation Timing

PPP Precise Point Positioning

PRN Pseudo-Random Noise

PVT Position, Velocity and Time

RAIM Receiver Autonomous Integrity Monitoring

RBF Radial Basis Function

RF Radio Frequency

RHCP Right-Hand Circular Polarization

RINEX Receiver Independent Exchange Format

RTC Real-Time Clock

RTK Real-Time Kinematic

SAS Signal Authentication Service

SBF Septentrio Binary Format

SCA Spreading Code Authentication

SCE Spreading Code Encryption

SDR Software-Defined Radio

SIS Signal In Space

SNR Signal-to-Noise Ratio

SPP Single Point Positioning

SQM Signal Quality Monitoring

SSR State Space Representation

SV Space Vehicle

SVM Support Vector Machine

TESLA Timed Efficient Stream Loss-Tolerant Authentication

TOW Time of Week

TTFAF Time To First Authenticated Fix

TTFF Time To First Fix

UBX u-blox Binary Protocol

WLS Weighted Least Squares

Chapter 1

Introduction

PNT (Position Navigation Timing) information delivered by GNSS (Global Navigation Satellite System) has become an implicit assumption of most transport and infrastructure operations. Maritime navigation is a representative case: position and timing feed voyage planning, electronic chart display, automatic identification of vessels and the synchronisation of shore-side services, so that a corrupted or denied PNT solution propagates directly into situational awareness and into the safety of navigation. The exposure of this dependency has grown faster than the countermeasures: interference events affecting civil maritime and aviation users are now reported regularly over entire sea regions [1], and the equipment required to produce them is no longer confined to specialised laboratories, since general-purpose SDR (Software-Defined Radio) platforms and open-source signal generators are sufficient to synthesise counterfeit GNSS signals [2]. Resilient PNT, understood as the ability of a navigation system to maintain a trustworthy solution, or to report that it cannot, under degraded and adversarial conditions, has therefore become a design requirement.

The vulnerability follows from the architecture of an open GNSS service. Signal structure, spreading codes and message format are public, so that any manufacturer can build an interoperable receiver, and the broadcast is one-way, so that a receiver cannot challenge a transmitter, negotiate a secret or request a retransmission [3]. A civil receiver consequently has no intrinsic means of establishing whether the signals it tracks originate from the declared satellites. Galileo addresses part of this gap with OSNMA (Open Service Navigation Message Authentication), a service that embeds authentication data in the E1-B navigation message so that a receiver can verify that the navigation data were transmitted by Galileo and were not modified [4, 5]. Two limits of that service define the problem addressed in this thesis. First, authentication applies to the navigation data and not to the physical signal, so that neither the denial of the signal by jamming nor the rebroadcast of authentic signals is covered by it. Second, OSNMA is normally consumed as a binary answer,

while the intermediate states that produce that verdict, such as the states of the key chain, the origin of each authentication tag or the use of previously stored material, are not exposed to the positioning engine, although they describe how the answer was reached and under which conditions it failed.

Detection of interference and spoofing has been studied predominantly inside the receiver, at signal level. Monitoring of the AGC (Automatic Gain Control) and of the noise floor, signal quality monitoring based on the shape of the function, multi-antenna processing and polarisation-based discrimination all achieve high detection performance, and machine-learning approaches applied to the acquisition surface follow the same principle [6, 7, 8, 9, 10, 11]. These techniques, however, require access to quantities that mass-market receivers do not expose, and in several cases dedicated hardware, which makes them difficult to deploy on the commercial equipment actually installed on board. A complementary line of work operates instead on the quantities that any receiver outputs, either through statistical processing of raw measurements [12] or through learning models applied to observables and receiver clock behaviour [13, 14, 15, 16]. On the cryptographic side, navigation message authentication has evolved from early proposals based on unpredictable signal content to the currently deployed service [17, 18, 19]. What the reviewed literature does not provide is the combination of the two: the internal states of the OSNMA protocol are not used as an observable quantity alongside the physical measurements, and the two monitoring layers are evaluated separately.

This thesis addresses that combination at the observation and PVT level, using low-cost commercial receivers and no access to the receiver internals. An OSNMA processing library [20] and a Galileo-only single point positioning solver [21] are integrated into a single pipeline that exposes every intermediate authentication state together with the observables reported by the receiver, and that admits or excludes satellites from the position solution according to their authentication status. The pipeline is characterised on static recordings collected in nominal and obstructed conditions, on live sessions with controlled start modes, and on a spoofing scenario generated through an SDR whose ground truth is known by construction. A data-driven detection stage is then built on the receiver observables alone, using approximately five days of labelled Galileo E1 data from the Jammertest 2025 campaign, and is evaluated on days that are never part of the training data. The main contributions are the pipeline and its diagnostic outputs, a feature set defined from physical mechanisms rather than from observation geometry, an evaluation protocol that assigns entire labelled transmissions rather than individual epochs to a split, and a systematic comparison of temporal models against memoryless baselines trained under identical conditions, which shows that the temporal context does not improve detection on this feature set.

The objectives in this work are the following.

- To integrate OSNMA processing and Galileo-only single point positioning into

one pipeline running on low-cost u-blox receivers (NEO-M8T, ZED-X20P), producing per-satellite and per-subframe authentication states aligned in time with the receiver observables.

- To characterise the behaviour of those states in reference recordings with open sky view and with approximately half of the sky obstructed, and to determine and interpret nominal behaviour.
- To quantify the effect of enforcing authentication on the position solution, by comparing the solution computed from all tracked Galileo satellites with the solution restricted to authenticated satellites.
- To measure the TTFAF (Time To First Authenticated Fix) across cold, warm and hot start modes over repeated static sessions.
- To assess the detection of anomalous reception from receiver observables only, with a model trained without attack labels, reporting ROC-AUC and F1 on campaign days excluded from training.
- To assess the classification of nominal, jamming and spoofing conditions from the same observables, over repeated splits assigned by transmission event, and to compare every model with a memoryless baseline trained under the same protocol.

Accordingly, the work answers the following research questions.

- How can the OSNMA authentication status be exploited at PVT level? Two policies are compared on the same recordings: computing the solution from authenticated satellites only, and computing it from all tracked satellites while excluding those whose authentication failed. What is the resulting effect on availability and on positioning accuracy in nominal and in obstructed reception conditions?
- Can the comparison between an OSNMA-enabled solution and an unauthenticated solution computed from the same measurements be used as a consistency check, and does the discrepancy between the two behave as a detection quantity in its own right?
- Are the observables reported by a low-cost single-frequency receiver sufficient to detect and to identify interference at the observation level, using anomalies of received power against elevation, receiver clock and Doppler behaviour, and dispersion across satellites?
- Does temporal dependence improve that decision with respect to an instantaneous one on the same features, and how do the resulting models compare with memoryless classifiers trained under an identical protocol?

- Do the intermediate states of the OSNMA protocol carry information that the observables cannot supply, and conversely? In particular, do the protocol states resolve events that the observables leave ambiguous, and are they therefore candidates for reducing the false alarms of an observation-level detector?

The thesis is organised in three parts. The first establishes the background required by the rest of the work: the GNSS fundamentals and the threats affecting civil receivers, and Galileo OSNMA together with the software architecture built around it. The second describes the experimental work, from the setup and the test scenarios to the methodology adopted for the observation-level detection models. The third reports and discusses the results obtained for both monitoring layers, and closes with the conclusions and the directions left open for future work.

Chapter 2

GNSS Fundamentals

2.1 Introduction

GNSSs provide PVT (Position, Velocity and Time) information by means of satellite signals received and processed by a user receiver [22, Ch. 1, pp. 1–6]. In this chapter, some fundamental GNSS concepts necessary for understanding the objectives of this thesis are presented: the constellations considered in this work, the structure of the Galileo E1 signal, the receiver processing chain that produces the observables, and the computation of a position solution from pseudorange measurements.

2.2 GNSS Constellations

Several global constellations are currently operational. They differ in orbital configuration, signal plan and services, but they share the same operating principle: satellites broadcast time-tagged navigation signals, and the receiver estimates its position and clock offset from the signals of multiple satellites in view. The experimental work of this thesis is carried out on GPS and Galileo only, and the description is therefore limited to these two systems.

GPS, operated by the United States, consists of satellites distributed across six orbital planes at an altitude of approximately 20,200 km, broadcasting on the L1, L2 and L5 bands; the L1 band is centred at 1,575.42 MHz. Galileo, developed jointly by the European Union and the European Space Agency, comprises more than 27 MEO satellites at approximately 23,222 km, broadcasting on the E1, E5 and E6 bands [23, 24]. The Galileo E1 band shares the same centre frequency as GPS L1, which makes the two systems interoperable at receiver level.

Galileo is the system of interest for this work, because OSNMA is transmitted on the E1-B component of its Open Service signal [4]. GPS is used in the controlled spoofing scenario of Chapter 6, where the counterfeit signals are generated on L1.

2.3 GNSS Segments

A GNSS can be described through three interacting segments [22, Sec. 2.1, pp. 7–18]. The space segment is the constellation itself: each satellite broadcasts ranging signals and navigation data, with onboard atomic clocks providing the timing reference, so that the receiver can estimate the signal propagation time and compute the satellite position at transmission time. The ground segment monitors the constellation, estimates orbits and clock corrections and uploads the navigation data to the satellites; for Galileo, it is also responsible for generating and distributing the authentication material required by OSNMA. The user segment consists of the receivers, which acquire and track the signals, demodulate the navigation data, form the observables and estimate the PVT solution; in an OSNMA-capable receiver an additional software module processes the authentication fields and attaches an authentication status to the navigation data. This subdivision is recalled here because the authentication service follows the same architecture, as described in Chapter 4.

2.4 Structure of a GNSS Signal

The signal broadcast by each satellite is built from a radio-frequency carrier, a spreading code and a navigation message [22, Sec. 2.2, pp. 18–38]. The *spreading code*, also called ranging code or PRN (Pseudo-Random Noise) code, is a binary sequence that is multiplied with the navigation data before transmission, while the *navigation message* carries the satellite clock corrections, the ephemeris describing the satellite orbit, and health and timing parameters.

Two properties of the PRN codes are relevant to the following chapters. The codes assigned to different satellites of the same constellation are nearly orthogonal, which allows several satellites to share the same carrier frequency, as in the CDMA (Code Division Multiple Access) scheme adopted by GPS and Galileo. In addition, a PRN code correlates strongly with its local replica only when the two are aligned in time, and the delay required for that alignment corresponds to the signal travel time, which is the quantity from which the pseudorange is derived in Section 2.6.3. The same correlation provides the gain needed to recover signals that reach the antenna well below the thermal noise floor, as detailed in Section 2.6.

2.5 The Galileo E1 Signal

The Galileo E1 signal is transmitted at 1,575.42 MHz and comprises three components [25, 26]. E1-A supports the Public Regulated Service, with encrypted ranging codes and navigation data reserved for authorised users; E1-B carries the

Open Service navigation data and is the component relevant to this thesis, since the OSNMA fields are inserted in reserved bits of the I/NAV message without modifying the basic open-service structure; E1-C is a ranging component without navigation data. The E1-B ranging code has a length of 4092 chips at a chipping rate of 1.023 Mchip/s, giving a repetition period of 4 ms.

The E1-B navigation message contains the satellite clock parameters, the ephemeris, and health and timing information, which the receiver needs in order to compute the satellite position and to correct the signal transmission time [25]. These are the data whose authenticity is assessed by OSNMA; the corresponding authentication fields are described in Chapter 4. All the recordings analysed in this thesis are collected in Galileo-only, single-frequency E1 configuration.

2.6 Receiver Signal Processing

Carrier, spreading code and navigation message are recovered by the receiver through a processing chain that turns weak radio-frequency signals into observables and navigation data. Only the steps that are relevant to the monitoring performed in this thesis are recalled here.

2.6.1 Front-End and Signal Acquisition

The antenna collects the composite signal broadcast by all satellites in view; GNSS signals are transmitted with right-hand circular polarization (RHCP), a property exploited by some anti-spoofing techniques discussed in Chapter 3. In the RF front-end the incoming signal is filtered, amplified and down-converted to an intermediate frequency, and an analog-to-digital converter (ADC) produces the digital samples used by the rest of the receiver. The gain applied before the converter is set by an automatic gain control stage, whose value is one of the quantities monitored in Chapter 8. Acquisition then searches over the two-dimensional space of code delay and Doppler frequency: a correlation peak above a detection threshold gives a coarse estimate of both quantities, which is handed over to the tracking stage for refinement.

2.6.2 Signal Tracking and Quality Monitoring

Tracking loops refine the carrier frequency, carrier phase and code delay estimates of each acquired satellite, keeping them aligned with the incoming signal as it evolves over time. Tracking also provides a direct measure of signal quality through the carrier-to-noise density ratio (C/N_0), expressed in dB-Hz as the power of the useful signal relative to the noise floor, and the signal-to-noise ratio (SNR), defined similarly but normalized to the receiver processing bandwidth [22, Sec. 4.2,

pp. 75–79]. These quantities are used throughout this thesis to characterise nominal signal conditions and the anomalies caused by interference or spoofing, as discussed in Chapter 3. Signal quality also depends on the observation geometry: signals received at low elevation travel a longer atmospheric path and are more exposed to multipath, which results in a lower C/N_0 and a noisier code-delay estimate. Receivers commonly apply an elevation mask to exclude low-elevation satellites, and weight the remaining measurements by elevation in the WLS solution, as discussed in Section 2.7.1.

2.6.3 Observables and Navigation Data

The main observable produced by the processing chain is the code pseudorange, obtained by comparing the time of the received PRN code with the locally generated replica:

$$R^j = c (t_r - t^j), \quad (2.1)$$

where c is the speed of light, t_r is the time of signal reception, measured in the time scale given by the clock of receiver r , and t^j is the time of signal transmission, measured in the time scale given by the clock of satellite j . Alongside the pseudorange, the receiver also produces the carrier phase and the Doppler observables. In parallel, it demodulates the navigation message, obtaining the satellite clock corrections, the ephemerides and additional parameters; from the ephemerides the satellite orbit is determined, which allows the satellite position at each transmission time to be computed, while the pseudoranges provide the measured distances between receiver and satellites.

The measurement is called *pseudo*-range because the two epochs of Equation (2.1) are read on two different and unsynchronised clocks, so the result differs from the true geometric range between satellite and receiver. Following [22, Ch. 4, pp. 65–67], the code pseudorange measured from satellite j can be modelled as

$$R^j = \rho^j + c (\delta t - \delta t^j) + T^j + \hat{\alpha}_1 I^j + TGD^j + M^j + \varepsilon^j, \quad j = 1, \dots, n, \quad (2.2)$$

where ρ^j is the geometric range between the satellite position at transmission time and the receiver position at reception time, δt and δt^j are the receiver and satellite clock offsets with respect to system time, T^j and I^j are the tropospheric and ionospheric delays, $\hat{\alpha}_1$ is the frequency-dependent factor scaling the ionospheric term, TGD^j is the satellite instrumental delay, and M^j and ε^j account for multipath and receiver noise.

Among these terms, the satellite clock offset, the instrumental delay and the ionospheric correction parameters are broadcast in the navigation message, while the tropospheric delay is obtained from a model; they can therefore be grouped into a single modelled quantity $D^j = -c \delta t^j + T^j + \hat{\alpha}_1 I^j + TGD^j$ and moved to

the measurement side. Neglecting multipath and receiver noise, Equation (2.2) becomes

$$R^j - D^j \simeq \sqrt{(x^j - x)^2 + (y^j - y)^2 + (z^j - z)^2} + c \delta t, \quad j = 1, \dots, n, \quad (2.3)$$

where (x^j, y^j, z^j) are the satellite coordinates, computed from the broadcast ephemerides, and the right-hand side contains the four unknown parameters: the receiver coordinates (x, y, z) and the receiver clock offset δt .

The receiver clock offset deserves particular attention, since it is the only one of the four unknowns that is not geometric. It arises because the receiver oscillator, typically a low-cost quartz, is not synchronised with GNSS system time and is affected by drift; unlike the atomic clocks onboard the satellites, whose offset is broadcast and can be corrected through D^j , it cannot be known in advance. Being common to all satellites tracked at a given epoch, it enters every measurement equation with the same value and must therefore be estimated together with the three receiver coordinates. At least $n \geq 4$ satellites in view are consequently required to solve the system of Equation (2.3), whose resolution is the subject of the next section.

2.7 Single Point Positioning

SPP (Single Point Positioning) is the standard GNSS code-based positioning mode in which the receiver computes its solution from its own GNSS measurements and the broadcast navigation data, without external corrections [27]. It is the mode used throughout this thesis, and it provides meter-level accuracy, since the error sources that are not compensated in its basic formulation, namely the residual ionospheric and tropospheric delays, the broadcast orbit and clock errors and multipath, remain in the solution. This basic PVT computation is the final consumer of the navigation data whose authenticity is assessed by OSNMA in this work, which is why the estimator is described here in the form in which it is later modified to account for the authentication status of each satellite.

The quantity to be estimated is the true state

$$\mathbf{x} = \begin{bmatrix} \mathbf{r}^T & b \end{bmatrix}^T, \quad (2.4)$$

where $\mathbf{r} = [x, y, z]^T$ is the three-dimensional receiver position in an Earth-centred, Earth-fixed frame and $b = c \delta t$ is the receiver clock offset expressed as a length. Given n pseudorange measurements collected at one epoch, the estimation problem consists in finding the state that best explains those measurements in a weighted least-squares sense. The estimator is a snapshot one: each epoch is solved independently from the others, with no filtering across epochs, so that the solution of one epoch cannot propagate an error into the next.

2.7.1 Weighted Least Squares Estimation

The WLS (Weighted Least Squares) approach is the method most widely adopted in SPP [22, Sec. 6.1, pp. 139–145]. Since the observation equations of Equation (2.3) are nonlinear with respect to the receiver position, they are linearized around a linearization point $\hat{\mathbf{x}}^k$, and the solution is obtained iteratively through a Gauss-Newton procedure.

In the implementation used in this thesis only the satellite clock correction is applied to the measurements, while the tropospheric and ionospheric terms of Equation (2.2) are left uncompensated and remain in the residuals; this is consistent with the single-frequency code-only configuration adopted throughout the experimental work.

At the first iteration the linearization point $\hat{\mathbf{x}}^0$ has to be initialized. In the implementation used in this thesis the position is initialized from the reference position of the recording, while the clock offset is initialized to zero at the first epoch of a recording and to the value estimated at the previous epoch afterwards. Given the current linearization point, the predicted geometric range to satellite i is

$$r_i = \|\mathbf{r}_{s,i} - \hat{\mathbf{r}}\|, \quad (2.5)$$

where $\mathbf{r}_{s,i}$ is the satellite position and $\hat{\mathbf{r}}$ is the position part of the current linearization point. The corresponding line-of-sight unit vector is

$$\mathbf{e}_i = \frac{\mathbf{r}_{s,i} - \hat{\mathbf{r}}}{r_i}. \quad (2.6)$$

It indicates the direction along which a change in receiver position produces the largest change in the computed geometric distance, and is used to build the design matrix of the linearized system. Since the signal requires a finite time to travel from the satellite to the receiver, during which the Earth continues to rotate, the geometric distance is further corrected for this effect, known as the Sagnac correction. The residual between the measured pseudorange and the value predicted from the current linearization point is

$$y_i = PR_i - \left(r_i + \hat{b} - c dt_{s,i} \right), \quad (2.7)$$

where PR_i is the measured pseudorange, $\hat{b}$ is the clock offset of the current linearization point expressed as a length, $dt_{s,i}$ is the satellite clock offset and c is the speed of light. The residual therefore measures how much the current estimate fails to explain the measurement, and is progressively reduced along the iteration.

The linearised observation model is built with a design matrix whose i -th row is

$$\mathbf{H}_i = \begin{bmatrix} -\mathbf{e}_i^T & 1 \end{bmatrix}. \quad (2.8)$$

Each row of $\mathbf{H}$ relates a change in the unknowns to the corresponding change in the predicted pseudoranges: the first three columns contain the negative line-of-sight unit vector, while the fourth column is a unit entry. The negative sign reflects the fact that a displacement of the receiver toward a satellite reduces the computed distance to that satellite by the same amount; a change in the receiver clock offset, instead, affects the computed distance to every satellite equally.

The weight matrix $\mathbf{W}$ accounts for the fact that the measurements do not all have the same error variance. It is taken as the inverse of the covariance matrix of the measurement errors, which reduces to a diagonal matrix of terms $1/\sigma_i^2$ when the errors affecting different satellites are assumed uncorrelated, so that noisier observables have less influence on the estimate. Since σ_i^2 is not directly observable, it is predicted from quantities available at the receiver. In SPP the satellite elevation angle θ_i is the most common choice, because signals received at low elevation travel a longer path through the atmosphere and are more exposed to multipath, and therefore exhibit a larger error variance. The elevation angle is thus not a weighting criterion per se, but is the parameter from which the variance is derived, through an elevation-dependent model of the form

$$\sigma_i^2 = \frac{\sigma_0^2}{\sin^2 \theta_i}, \quad (2.9)$$

where σ_0 is the zenith-referred standard deviation of the pseudorange error, so that the weight assigned to each measurement reduces to $\sin^2 \theta_i$ and the variance grows as the elevation decreases.

The weighted least-squares correction to the current linearization point is then

$$\Delta \hat{\mathbf{x}} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{y}, \quad (2.10)$$

with $\Delta \hat{\mathbf{x}} = [\Delta x, \Delta y, \Delta z, \Delta b]^T$. The linearization point is updated according to

$$\hat{\mathbf{x}}^k \leftarrow \hat{\mathbf{x}}^{k-1} + \Delta \hat{\mathbf{x}}, \quad (2.11)$$

and the geometric distances, the line-of-sight vectors and the design matrix are recomputed with respect to the updated point. As the estimate approaches the solution, the magnitude of the correction decreases at each successive iteration. The procedure is considered to have converged when the norm of the position correction falls below a predefined threshold, typically at the sub-millimeter level, and the resulting $\hat{\mathbf{x}}^k$ is taken as the final snapshot position and clock estimate $\hat{\mathbf{x}}$ for that epoch. If convergence is not reached within a maximum number of iterations, or if fewer than four satellites are available, the measurements of that epoch are discarded so no solution is computed for it.

Chapter 3

GNSS Security

3.1 Introduction

The previous chapter described how a GNSS receiver obtains a navigation solution from public radio signals and broadcast navigation data. This openness is essential for civilian accessibility, but it also means that a conventional receiver has limited means to distinguish an authentic signal from a counterfeit or manipulated one.

Resilient PNT denotes the ability of a system that relies on position, navigation and timing information to prevent the disruption of its PNT data, to detect a disruption when it occurs, and to respond to it by maintaining a usable solution or by declaring that none is available [28]. Resilience is therefore a property of the whole PNT architecture, and it requires the disruption to be observable.

This chapter summarises the threats affecting a civil receiver, defines detection and mitigation and the level at which each operates, and reviews the available countermeasures, ending with the cryptographic ones, to which Galileo OSNMA belongs.

3.2 GNSS Threats

GNSS receivers may be affected by natural propagation effects, radio-frequency interference, hardware faults, or deliberate attacks. The threats most relevant to this thesis are those that can alter the measurements or the navigation data used by the receiver.

- **Unintentional interference** is any external radio-frequency emission that degrades the useful GNSS signal without being aimed at it, for example the harmonics or the out-of-band emissions of broadcasting systems such as DVB-T falling close to GPS L1 or Galileo E1[29, 3].

- **Jamming** is intentional interference aimed at preventing the receiver from acquiring or tracking GNSS signals. A jammer does not attempt to produce a valid navigation solution: it denies the service by raising the noise level or by saturating the receiver front-end. When the jammer is aligned with the target GNSS band, it reduces the carrier-to-noise density ratio of the received signals, and if the interference power is high enough the receiver loses the tracking loops[6, 3].
- **Meaconing** consists of receiving and rebroadcasting authentic GNSS signals with an additional delay. Since the rebroadcast signal carries valid navigation data, the receiver interprets the delay as additional propagation time and computes an incorrect position or time solution[30].
- **Spoofing** is the generation or retransmission of counterfeit GNSS signals intended to make the receiver compute a false PVT solution. Compared with jamming, spoofing is harder to recognise, because the receiver continues to operate while using manipulated measurements or data[2, 31, 12].
- **Tampering** refers to attacks requiring physical access to the receiver. In the context of authenticated GNSS this is relevant because trusted keys and cached authentication material are stored inside the receiver[5].

Comparable effects can also be produced downstream of the receiver, for example by manipulating the reported position or the communication channel used to transmit it. These cases belong to general cyber-security and are not addressed here. This chapter focuses on upstream threats, where the disturbing signal reaches the receiver antenna and can therefore be addressed by signal monitoring or by authentication.

3.3 Detection and Mitigation of GNSS Interference

Detection is the ability to recognise that a signal, a measurement or a navigation-data set is not consistent with the expected behaviour of an authentic transmission[3, 2]. Detection can be performed at different levels of the receiver chain, and the level determines both the information available and the equipment required:

- at **signal level**, on the digitised samples or on the correlator outputs, for instance through the distortion of the correlation peak;
- at **receiver level**, on internal quantities such as the automatic gain control setting or the tracking loop state;

- at **observation level**, on the observables reported by the receiver, such as the carrier-to-noise density ratio, the pseudoranges and the Doppler measurements;
- at **PVT level**, on the estimated solution and on its residuals, or on the cryptographic verification result attached to the navigation data.

The output of a detection function is an alarm, a score or an authentication label. The work presented in this thesis operates at observation and PVT level, since these are the only quantities available on the mass-market receivers considered here.

Mitigation is the action taken after detection in order to provide a trustworthy PVT solution, or to declare that no trustworthy solution can be computed. It may consist of excluding a satellite, discarding a measurement, switching to another sensor, applying spatial filtering or declaring the solution unavailable[32, 33].

The distinction between the two functions is particularly relevant for OSNMA. The service verifies whether the received navigation data are authentic, and therefore provides detection, but it does not prescribe how the PVT engine should treat the data of a satellite whose authentication has failed or has not been completed. That choice belongs to the receiver implementation and to the application policy: discarding every satellite that is not authenticated increases the confidence in the solution, but it also reduces the number of usable measurements and may leave the receiver without a solution at all, particularly when the authentication fails for reasons unrelated to an attack, such as a partially obstructed sky view or the loss of the key chain. Availability and trustworthiness are therefore in competition, and the trade-off between them cannot be resolved by the authentication service alone.

One step of this thesis operates exactly at this interface. The positioning engine described in Chapter 4 allows the authentication status of each satellite to be enforced or ignored, so that the same measurements can be processed under both policies. The objective is to quantify how the use of the detection result affects the availability and the accuracy of the position solution, in nominal conditions as well as under the attacks considered in this work.

3.4 Anti-Spoofing and Anti-Jamming Techniques

GNSS protection techniques can be grouped into non-cryptographic monitoring methods and cryptographic schemes. The first group uses properties already available in the received signal or in the receiver observables. The second group modifies the signal or the navigation message by adding authentication or encryption mechanisms.

3.4.1 Non-Cryptographic Detection Parameters

Non-cryptographic techniques do not require a modified GNSS signal. They search for inconsistencies in receiver observables or in signal-domain quantities, compared against expected values and thresholds. Typical examples include:

- **Received power and AGC monitoring:** spoofing typically raises the received signal power, which drives the receiver AGC (Automatic Gain Control) to reduce its gain in order to avoid saturating the ADC; monitoring these gain variations can therefore reveal an attack[7, 6].
- **Time-of-arrival and pseudorange consistency:** spoofed signals are rarely perfectly consistent with the authentic constellation geometry, so comparing the pseudoranges, which are derived from the arrival times, across satellites can reveal inconsistent measurements. This is the basis of RAIM (Receiver Autonomous Integrity Monitoring); when an inconsistent measurement is not only flagged but also removed from the position computation, the process is known as FDE (Fault Detection and Exclusion)[34, 35].
- **SQM (Signal Quality Monitoring):** operates on the correlator output rather than on PVT-level observables, looking for a distorted or split correlation peak. A spoofing attempt can produce a secondary correlation peak alongside the authentic one, which MPD (Multi-Peak Detection) techniques are designed to identify; when the spoofed and the authentic delays are too close to be resolved as separate peaks, monitoring the shape of the combined peak for distortion remains a useful indicator[36, 8].
- **Angle or direction of arrival:** exploits differential carrier-phase measurements between multiple antennas. Since a single ground-based spoofer generates all its counterfeit signals from the same physical location, they arrive from the same direction, unlike genuine signals from satellites spread across the sky[9].
- **Signal polarization:** authentic signals are transmitted with a pure RHCP polarization (Section 2.6.1), and acquire a degraded LHCP component only through multipath reflections that differ satellite by satellite; the signals of a spoofer, following a single path from transmitter to receiver, tend to share an unnaturally consistent polarization signature across all counterfeit satellites[10].
- **Inertial measurements:** tightly coupling GNSS with an IMU (Inertial Measurement Unit) provides an independent estimate of motion that does not rely on radio signals; a divergence between the GNSS-derived trajectory and

the inertial one is a detection parameter, and the inertial solution can also serve as a temporary mitigation while the GNSS solution is suspect[37, 38].

These techniques can be effective, especially when combined, but they are not cryptographic proofs of authenticity: they infer an attack from a physical inconsistency, and their performance depends on the receiver hardware, on the environment and on the sophistication of the attacker. Several of them also require access to signal-level or receiver-level quantities, or additional hardware such as multiple antennas or an inertial unit, which restricts their applicability on mass-market equipment, where the cost and the complexity of the countermeasure must remain compatible with the expected threat level.

3.4.2 Cryptographic Authentication and Encryption

The idea of authenticating GNSS signals is not recent. Early proposals suggested exploiting unpredictable information in the signal or in the navigation data, so that a spoofer unable to predict it could be distinguished from an authentic transmitter[17]; these were followed by schemes based on secret spreading sequences and by navigation-message-only authentication[18], the latter still vulnerable to replay attacks[19]. Galileo has since taken a lead among the GNSS providers by including a navigation message authentication service in its Open Service baseline, which reached Initial Service Declaration in 2025 [4].

Cryptographic approaches introduce information that an attacker is not able to forge. They are classified according to the part of the signal they protect and to whether they authenticate or encrypt it:

- **NMA (Navigation Message Authentication)** verifies that the navigation data were generated by the system provider and have not been modified.
- **SCA (Spreading Code Authentication)** applies the same principle to the ranging code, by introducing unpredictable or later-verifiable code components.
- **NME (NAavigation Message Encryption)** restricts access to the navigation data to authorised users holding the appropriate secret material.
- **SCE (Spreading Code Encryption)** restricts access to the ranging code in the same way. Military signals such as the GPS P(Y) code already implement this protection, permanently reserving the ranging code to authorised government and military receivers.

Authentication and encryption have different security effects. Authentication keeps the signal open and provides a verification result after reception, and is therefore mainly a detection mechanism. Encryption prevents the unauthorised

generation or use of the protected content, and therefore acts as a preventive form of mitigation. Civil open signals historically offered neither property, which is why modern GNSS services increasingly include authentication mechanisms.

3.4.3 Open Service Navigation Message Authentication

Galileo OSNMA is an NMA service transmitted on the E1-B Open Service signal [39, 4]. It authenticates selected Galileo navigation data by adding cryptographic material to the I/NAV message, while keeping the signal publicly accessible. It is based on TESLA, a delayed-disclosure authentication protocol in which the tags are transmitted first and the key needed to verify them is disclosed later [40, 39, 5]. A spoofer can still receive the open signal and decode the navigation message, but it cannot generate valid authentication tags without the corresponding cryptographic material.

OSNMA provides data-level authentication. It indicates whether the navigation data used by the receiver are consistent with the Galileo authentication material, but it does not authenticate the ranging measurement itself, and it does not exclude a satellite from the PVT computation. It is therefore a detection layer added on top of the open signal [4], and this is precisely the property exploited in this thesis: the outcome of the verification, together with the intermediate states that produce it, is treated as an additional observable of the receiver and combined with the physical ones.

3.4.4 Signal Authentication Service

Galileo SAS extends authentication to the ranging domain by introducing temporarily unpredictable elements in the E6-C signal [26]. The receiver records the signal component and verifies it after the corresponding information is disclosed. This protects against attacks that preserve valid navigation data but manipulate the apparent signal arrival time, such as some replay or meaconing scenarios, because the ranging component itself becomes part of the authentication process [41].

3.4.5 Chips-Message Robust Authentication

CHIMERA is a GPS L1C authentication concept, developed by the MITRE Corporation and tested by the US Space Force on GPS III satellites, that combines navigation message authentication and spreading code authentication [42]. At the message level it signs the C/NAV navigation data with a digital signature, following the same principle as NMA. At the code level it relies on a watermarking scheme: short encrypted markers are inserted into the L1C spreading code at unpredictable positions and disclosed later, allowing the receiver to verify after the fact that the

code it tracked matches the authentic one. Unlike OSNMA, CHIMERA has not reached operational service status and remains a demonstration capability.

3.5 Summary

Among the countermeasures reviewed in this chapter, the non-cryptographic ones operate on quantities that are either internal to the receiver or affected by the environment, and they infer an attack from a physical inconsistency, without ever proving the origin of the signal. Cryptographic authentication provides that proof for the navigation data, and OSNMA is the only such service currently operational on a civil open signal, which makes it the element of the architecture on which this work is built. Its protocol, its data structure and the receiver workflow it imposes determine which states a receiver can observe and when, and are therefore described in detail in the following chapter, before they are used as a detection layer in the experimental part of the thesis.

Chapter 4

Galileo OSNMA: Principles, Architecture, and Implementation

4.1 Introduction

Chapter 3 introduced OSNMA as the Galileo Navigation Message Authentication service transmitted on the E1-B Open Service signal. This chapter describes it in the detail required by the rest of the thesis, and does so in three steps. Section 4.2 defines the data structures that carry OSNMA in the SIS (Signal In Space). Section 4.3 uses those structures in the order in which a receiver retrieves, buffers and verifies them. Section 4.4 describes the software architecture implemented in this work, which turns that workflow into the diagnostic outputs analysed in the experimental chapters.

Only the elements needed to interpret the results are reported here; the complete field-by-field definition of the protocol is given in the official interface documents [39, 5, 4], to which the reader is referred for the bit-level layout of each message.

4.2 OSNMA Data Structures

OSNMA uses previously reserved bits of the Galileo E1-B I/NAV message [25, 39]. This preserves backward compatibility: receivers that are not OSNMA-capable ignore these fields, while OSNMA-capable receivers decode them and use the additional cryptographic material to authenticate selected navigation data. Not every satellite necessarily transmits OSNMA at a given time; when it does not, its navigation data can still be authenticated indirectly through the material

broadcast by another satellite, a mechanism called cross-authentication that is used operationally in Section 4.3.

4.2.1 OSNMA Field Structure

E1-B									
Even/odd=1	Page Type	Data j (2/2)	OSNMA	SAR	Spare	CRC _j	SSP	Tail	Total (bits)
1	1	16	40	22	2	24	8	6	120
Even/odd=0	Page Type	Data k (1/2)						Tail	Total (bits)
1	1	112						6	120

Figure 4.1: Structure of the Galileo E1-B I/NAV page and sub-frame, showing the position of the 40-bit OSNMA field within each page[39].

As shown in Figure 4.1, every Galileo E1-B I/NAV page carries a 40-bit OSNMA field. Since an I/NAV page is transmitted every two seconds, a 30-second sub-frame contains 15 such fields, for a total of 600 OSNMA bits [39]. Each field is split in two: the first 8 bits contribute to the HKROOT message and the remaining 32 bits to the MACK message, so that over one sub-frame the receiver obtains a 120-bit HKROOT message and a 480-bit MACK message. The HKROOT part carries the service status and the DSM blocks, while the MACK part carries the authentication tags, the Tag-Info fields and the TESLA keys.

4.2.2 The HKROOT Message

The HKROOT message contains the NMA Header followed by the DSM (Digital Signature Message) field [39]. The DSM field is transmitted block by block, so a complete DSM may span several sub-frames, and its blocks may also be received from different satellites: the receiver groups them by DSM ID and orders them by Block ID before reconstruction.

NMA Header

The NMA Header is the first field of every HKROOT message and describes the status of the service and the chain context through three sub-fields [39, 4]. The 2-bit NMA field reports the overall service status, with the receiver interpretation given in Table 4.1. The 2-bit CID field identifies the TESLA chain currently associated with the transmitted material, which allows the receiver to distinguish chain instances during a transition. The 3-bit CPKS field reports the status of the public key and of the TESLA chain, and takes the values nominal, end of chain, chain revoked, new public key and public key revoked, the remaining values being reserved. Read together, NMA and CPKS tell the receiver whether it is in nominal operation, in a renewal or end-of-chain condition, or in a revocation condition.

Table 4.1: NMA values and receiver interpretation.

Status	Receiver interpretation
Test	Processing may be executed, but the output shall be treated as test data.
Operational	Successfully verified data may be provided as operational OSNMA output.
Don't Use	Affected OSNMA material shall not be used for authentication.

Digital Signature Messages

The DSM field carries either a DSM-PKR, used for public-key renewal, or a DSM-KROOT, used to distribute a signed TESLA root key together with the parameters of its chain [39]. Both types are transmitted as sequences of 104-bit blocks, each preceded by a DSM Header that identifies the message through the DSM ID and the position of the block inside it through the Block ID.

DSM-PKR and the Merkle tree. The DSM-PKR message carries a new public key together with the information required to authenticate it against the OSNMA Merkle tree [39, 43]; that public key is then used to verify the digital signature of DSM-KROOT messages. The Merkle tree of Figure 4.2 has 16 leaves, each corresponding to one possible public key or service message and obtained by hashing the concatenation of the key type, its ID and the key itself; the tree is built by repeatedly hashing pairs of nodes up to a single root, which is the element the receiver trusts. A DSM-PKR provides one leaf, identified by the MID field, and the intermediate nodes needed to reconstruct the path to the root, carried in the ITN field, alongside the type, identifier and value of the new key. If the root recomputed from that path matches the trusted root, the public key is accepted.

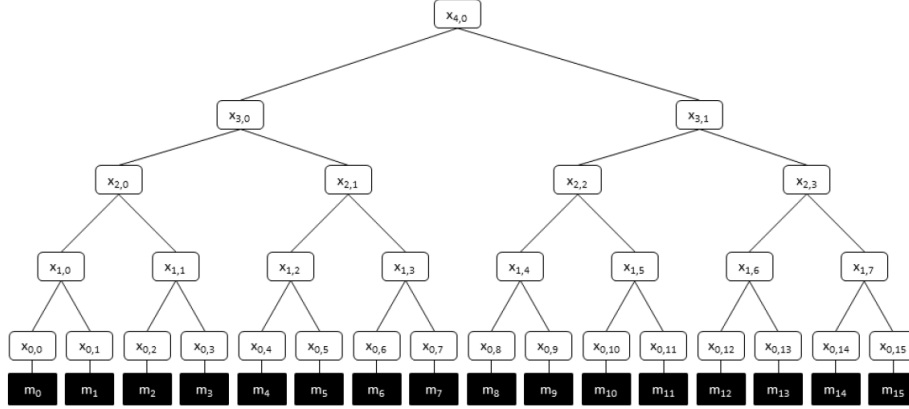


Figure 4.2: Structure of the Merkle tree used by OSNMA[39].

DSM-KROOT. The DSM-KROOT message carries the TESLA root key, named KROOT, the parameters defining the chain to which it belongs, and a digital signature that allows the receiver to authenticate the KROOT with a trusted public key [39, 5]. Besides the KROOT value and the signature, its fields specify the public key to be used for verification, the Chain ID, the hash and MAC functions, the key and tag sizes, the time of applicability, the MAC Lookup Table and the chain-specific pattern α .

Two of these parameters matter for the rest of the chapter. The MAC Lookup Table selects, from a predefined set of sequences, which ADKD type is associated with each tag position in the MACK section: some positions are fixed for the whole duration of the chain, while others are flexibly assigned and protected through MACSEQ. The pattern α is not a secret stored by the receiver, but a chain-specific parameter that enters the TESLA chain computation and differentiates one chain instance from another.

4.2.3 The MACK Message

The MACK message is transmitted in parallel with the HKROOT message and carries the material used to authenticate navigation data [39]. A MACK block is composed of a MACK Header, a sequence of Tags&Info fields and a TESLA Key field.

The MACK Header describes the first tag of the block, protects flexible tag assignments and indicates the freshness of the authenticated data. Tag₀, the first tag of the first MACK block in a sub-frame, is associated with self-authentication: it authenticates data from the same satellite that transmits it, unlike the other tags of the block, which may authenticate a different satellite through cross-authentication. The MACSEQ field allows the receiver to check that the Tag-Info assignments of

the flexible positions were not altered. The COP (Cut-Off Point) field indicates how many 30-second sub-frames have elapsed since the authenticated navigation data last changed, which helps the receiver decide whether stored navigation data are still compatible with a newly received tag.

Tags, Tag-Info and TESLA Key

After the header, the MACK block contains pairs of Tag and Tag-Info fields, each pair representing one authentication event. The Tag-Info field identifies the data authenticated by the associated tag: the PRN of the satellite whose data are authenticated, the ADKD identifying the type of data, and the Data COP [39]. Each Tag field is a truncated MAC, whose meaning is fixed by the adjacent Tag-Info field and by the MAC Lookup Table of the chain in force; its verification is described in Section 4.3.4.

The block closes with the Key field, which carries a disclosed TESLA chain key [39]. TESLA keys belong to a one-way chain rooted in the KROOT value authenticated through DSM-KROOT, so that a disclosed key can always be traced back to an already trusted chain element, as described in Section 4.3.4.

Figure 4.3 collects the structures described above into a single view: the OSNMA field inside each E1-B page, its split into HKROOT and MACK messages, the DSM messages reconstructed from the former, and the Tags&Info pairs and TESLA key carried by the latter. These are the objects available to the receiver in the SIS; the next section uses them in their operational order.

4.3 OSNMA Receiver Workflow

4.3.1 Receiver Requirements

OSNMA requires two receiver capabilities that a conventional unauthenticated GNSS solution does not need: a bounded time reference and enough memory to retain data until the delayed TESLA material becomes available [5, 40].

Time Requirements

The receiver holds a local time reference, estimated from the received signals, which shall be loosely synchronised to GST (Galileo System Time) before OSNMA information is processed [5]. This requirement preserves the TESLA security condition: the receiver must be able to establish that the navigation data and the corresponding tag were received before the associated chain key was disclosed. The synchronisation requirement is denoted by T_L and equals 30 s.

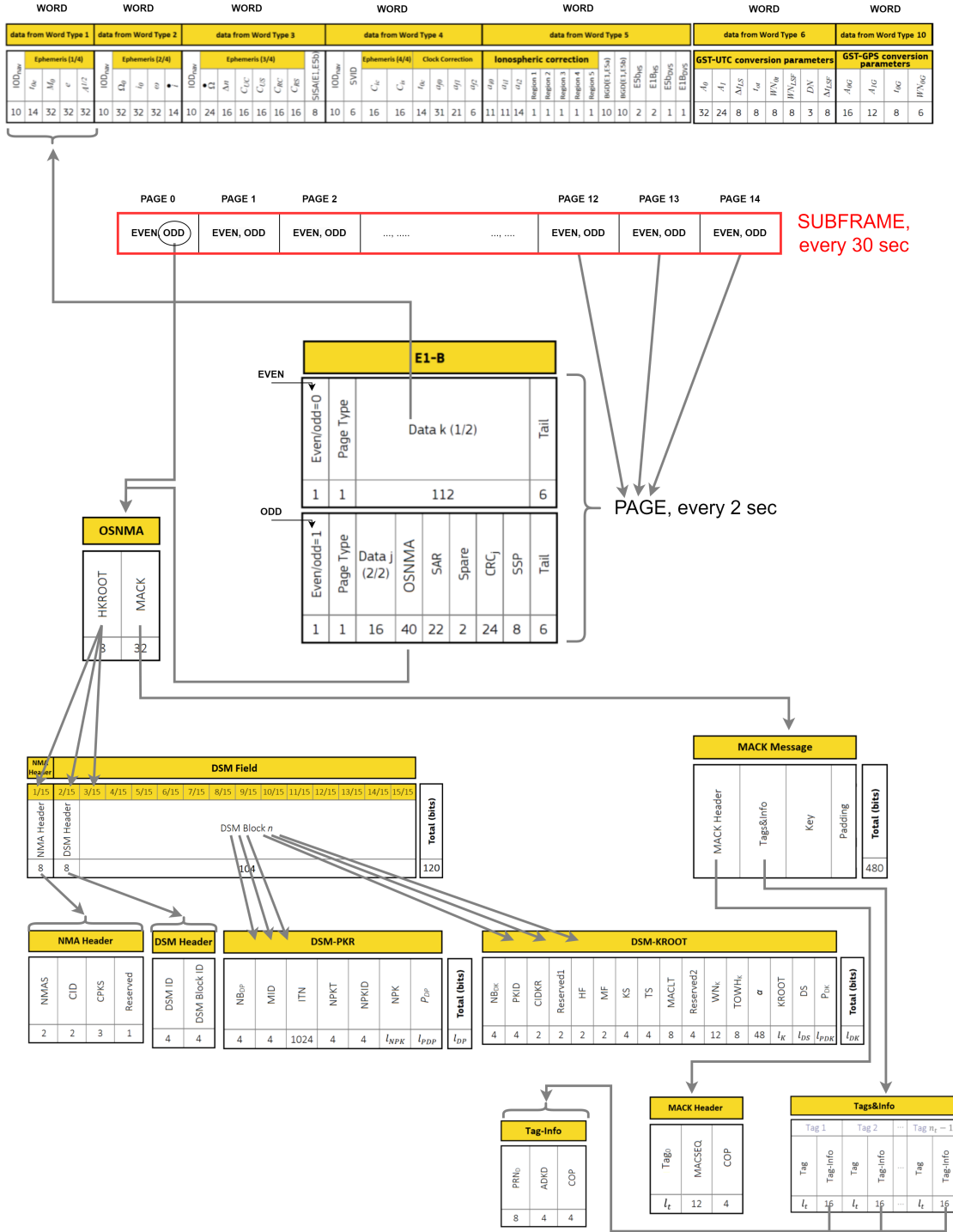


Figure 4.3: Overall OSNMA structure. Author's elaboration synthesized from the data structures defined in [39].

At start-up the receiver is in one of two states. Either it already has a local estimate of GST, denoted GST_{Rx} , together with its uncertainty B , typically because an estimate obtained in a previous session was propagated by an internal RTC while the RF front-end was powered down; in that case B accounts for the drift accumulated since the last reliable synchronisation, and processing can start if $B < T_L/2$. Or it has no usable time information, and an estimate of GST and of its uncertainty must be obtained beforehand, for instance from an external clock or, for connected receivers, from a secure time-transfer source accurate enough with respect to T_L [5, 40]. In either case, a time solution computed only from unauthenticated navigation data should not be used blindly to reduce B , since that solution is not itself authenticated.

Memory Requirements

The receiver must store both long-lived trust material and temporary operational material [5]. Non-volatile memory holds the provisioned or already trusted elements, such as the Merkle Tree Root, the public keys, the KROOTs and their applicability metadata. Volatile buffers hold incomplete DSM messages, MACK blocks, candidate TESLA keys, pending tags, Tag-Info fields and navigation data waiting for authentication. The design rule is that a received object should not be discarded while a valid authentication path may still refer to it, which matters precisely because TESLA delays key disclosure by construction.

4.3.2 OSNMA Data Retrieval

Retrieval means that the receiver collects and stores an object; it does not imply that the object is already authenticated. Only a small subset of the OSNMA material can be provisioned from the Galileo ground segment, for example through the GSC OSNMA server: the Merkle Tree Root and the public key. The KROOT, the disclosed TESLA keys, the tags, the Tag-Info fields and the navigation data are retrieved cyclically from the SIS and must be buffered until the corresponding verification step can be performed.

Trust Anchor and Public Key

The Merkle Tree Root is the stable trust anchor used to authenticate the public keys transmitted through DSM-PKR [39, 43]. It is not reconstructed from the SIS during normal operation: the receiver obtains it through a trusted provisioning process, typically from the GSC OSNMA server or from a configuration produced from it [44]. If the root is missing or not trusted, the receiver cannot authenticate a new public key received from the SIS, although it may still use a previously authenticated key that is valid for the current service status.

A public key can enter the receiver state in three ways, and the distinction matters because only one of them requires Merkle-tree verification in the current acquisition interval. A key provisioned through the GSC OSNMA server is external to the SIS reconstruction process and can be used directly to verify DSM-KROOT messages, provided that its identifier and status context are coherent. A key retrieved from the SIS is reconstructed from DSM-PKR blocks, grouped by DSM identifier and ordered by Block ID, and is stored only as a candidate until it is verified against the trusted root; while collecting those blocks the receiver also monitors the NMA Header, since NMA and CPKS determine whether public-key material may be used, renewed or discarded, even though they authenticate nothing by themselves. Finally, a key authenticated in a previous session or installed through trusted provisioning can be reused, which shortens start-up unless the current NMA status indicates renewal, revocation or mismatch.

TESLA Root Key

The TESLA root key is obtained from a DSM-KROOT reconstructed from the SIS, assembled as for DSM-PKR from blocks transmitted over several 30-second sub-frames [39]. At retrieval time it is still only received material, and becomes trusted only once its digital signature has been verified with a trusted public key. While waiting for that verification the receiver can already buffer MACK data, tags, Tag-Info fields and candidate TESLA keys indexed by GST and Chain ID, so that once the KROOT becomes trusted the buffered material whose time association is still valid can be processed immediately.

Cyclic SIS Data Retrieval

During normal operation the receiver repeats the same retrieval task at each 30-second sub-frame [5], collecting the OSNMA material from the MACK message and the corresponding navigation data from the ordinary I/NAV words, and storing each object with its satellite, data type, Chain ID and GST association. The cyclic retrieval mainly involves:

- **TESLA chain keys**, stored as candidates until they are linked to a trusted KROOT or to a previously verified chain key;
- **tags and Tag-Info fields**, stored with the information required to interpret them, namely the transmitting satellite, the tag position and the GST sub-frame;
- **navigation data**, stored by satellite, data type, issue of data and reception time.

For authentication, the receiver can use navigation data received before the applicable tag or in the same sub-frame as that tag. Data reconstructed only after the tag was received cannot replace the data that should have been stored for the tagged interval.

4.3.3 Chain of Trust and Start Modes

The objects described above form a single chain of dependencies: none of them can be used before the one it relies upon has been verified. Figure 4.4 summarises that chain, and Figure 4.5 shows the corresponding processing flow. A public key must be trusted before a DSM-KROOT signature can be accepted; a verified KROOT, or a previously verified chain key, must be available before a disclosed TESLA key can be accepted; a verified TESLA key is required before a pending tag can be checked; and a successful tag verification is required before the associated navigation data can be marked as authenticated [5]. From a temporal point of view, the receiver first establishes or recovers this chain of trust and then enters the cyclic processing loop.

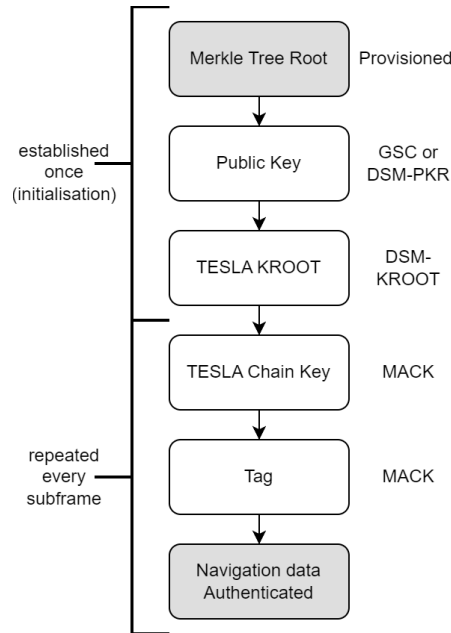


Figure 4.4: High-level OSNMA chain of trust. Each element can only be used once the previous one has been verified.

How much of that chain has to be established depends on the material already available in the receiver, which defines three start modes:

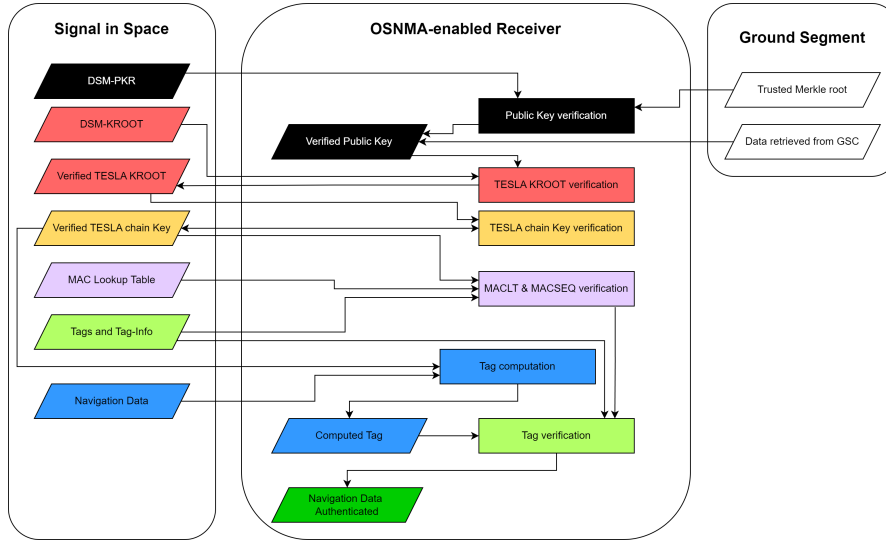


Figure 4.5: Processing of the OSNMA authentication material.

Cold start the receiver has no usable OSNMA operational material, apart from the possibility of loading the Merkle Tree Root, and must establish the whole trust path: load the root, obtain and verify or select a public key, and retrieve and verify a KROOT. During this interval navigation data and MACK material may be buffered, but cannot yet be declared authenticated.

Warm start the receiver already holds a verified public key but still has to acquire or confirm the KROOT of the active chain, and can therefore skip public-key acquisition unless the monitored NMA status indicates renewal, revocation or mismatch.

Hot start the receiver restarts with a coherent state already available: a trusted public key, a trusted KROOT or a recent verified TESLA key, and a local time estimate accurate enough to place the next sub-frame in the expected GST interval. If the NMA Header, the Chain ID or the received GST are inconsistent with the stored state, the receiver falls back to a warm or cold start.

4.3.4 Processing of the Authentication Material

The processing of the authentication material is the set of checks performed once enough OSNMA objects have been retrieved to continue the trust chain. Not every check runs at every sub-frame: some are triggered only when a complete DSM-PKR or DSM-KROOT becomes available. They are presented together because they belong to the same workflow.

NMA Status Monitoring

At each sub-frame, before using received or stored material, the receiver extracts the NMA Header and evaluates NMAS and CPKS jointly [39, 4]. Their combined interpretation indicates nominal operation, test operation, public-key renewal, chain renewal, end of chain, revocation, or a condition in which the material shall not be used. The receiver compares the current status with the one used for previous authentication decisions: if it is stable the loop continues, while a transition changes the set of objects that may be accepted. An announced future chain, for instance, may require collecting a new KROOT while data are still authenticated with the current chain; a public-key renewal may require collecting and verifying a DSM-PKR; a revocation requires the affected material and the dependent pending data to be invalidated.

Operational status allows verified navigation data to be provided as OSNMA-authenticated output, but it never replaces a cryptographic check. Test status allows the same processing chain to run, with the result labelled as test data. Don't Use and revocation prevent the authentication label from being assigned to data depending on the affected material, although an unauthenticated GNSS solution may still be computed.

Verification of the Public Key and of the TESLA Root Key

A public key reconstructed through DSM-PKR becomes trusted only if the receiver can link it to the locally trusted Merkle Tree Root [39, 5]. The receiver takes the candidate key, its identifier and the intermediate nodes transmitted with it, reconstructs the path from the corresponding leaf to the root, and compares the computed root with the trusted one. On success the key is stored with its identifier and validity context; on failure it is rejected, and the DSM-KROOT messages depending on it cannot be accepted. This SIS-specific verification is not repeated for a key obtained from the GSC OSNMA server or from a previous trusted session.

A KROOT received through DSM-KROOT is verified by checking the digital signature of the reconstructed message with the trusted public key it identifies, together with the coherence of the Chain ID, of the applicability time and of the current NMA status [5]. A successful verification authenticates not only the KROOT value but the whole chain context, namely the hash and MAC functions, the key and tag sizes and the MAC Lookup Table, which are then stored as the active or upcoming chain context. If the signature fails, the public-key identifier is unknown or the status context is inconsistent, the KROOT is not accepted.

Determination and Verification of the GST Sub-frame

The GST sub-frame, GST_{SF} , is the temporal label that connects the objects processed in the same authentication interval. It serves three purposes: for a TESLA key it determines the position in the chain, for a tag it identifies the authentication interval and enters the computation of the expected tag, and for navigation data it records when they were extracted from the I/NAV message, which decides whether they can be matched with a given tag.

The GST_{SF} is derived from GST_{SIS} , itself retrieved from the SIS, for example from I/NAV word type 5 or word type 0 [5]. Before using it, the receiver checks that GST_{SIS} is consistent with its local estimate GST_{Rx} : since $T_L = 30$ s and GST_{Rx} is treated as the centre of the receiver time estimate, with the uncertainty considered on both sides, the difference between the two must be smaller than $T_L/2$. This prevents the receiver from using a SIS time value too far from its local reference to safely associate MACK data with the correct disclosure interval. For the check to be meaningful, GST_{Rx} cannot be obtained solely by tracking or decoding the same signal under validation: an attacker replaying an entire old sub-frame, timestamp included, would shift GST_{SIS} and GST_{Rx} together and pass the check undetected. GST_{Rx} must therefore originate from, or be periodically anchored to, a time reference independent of the signal being authenticated.

If the check is satisfied, GST_{SIS} is mapped to the corresponding 30-second sub-frame, and all I/NAV pages and MACK data belonging to that interval receive the same label GST_{SF} , which is the start time of the interval rather than the exact GST_{SIS} value. Once an initial GST_{SF} has been derived and verified, the receiver may propagate it internally by adding 30 s at each new sub-frame, provided that the sub-frame sequence is continuously tracked.

Verification of the TESLA Chain Key

Once a KROOT or another chain key has been authenticated, the receiver can verify newly disclosed TESLA keys [5, 40]. Keys of the same chain are linked by a one-way function F : a previous key can be reconstructed from a later disclosed key, but future undisclosed keys cannot be computed, which is what prevents an attacker from deriving future keys and producing valid tags for a future sub-frame in advance.

At each step the receiver hashes together the disclosed key, the GST_{SF} of the key being reconstructed and the pattern α . The GST_{SF} binds the computation to the correct temporal position in the chain, while α acts as a chain-specific diversifier that prevents the hash relation between keys, and any generic precomputed relation, from being reused across chains. If K_i is the candidate key and K_j a previously authenticated key, with $j < i$, the receiver applies F the required number of times and checks whether

$$K_j = F^{i-j}(K_i),$$

as illustrated in Figure 4.6. If the result matches the trusted key, the new key becomes trusted, can be used to verify pending tags and can itself serve as a reference for later verifications.



Figure 4.6: Recursive verification of a TESLA chain key against a previously authenticated key [5].

Tag Verification and Authentication of Navigation Data

The receiver authenticates navigation data by identifying an applicable tag, checking its consistency with the MAC Lookup Table, selecting the applicable verified TESLA key and recomputing the expected tag [5]. Navigation data retrieved in a given interval are linked to tags transmitted according to the ADKD type: clock and ephemeris data associated with ADKD 0 and retrieved in the sub-frame at time GST_{SF} , for example, are authenticated by tags transmitted in the following sub-frame, so different data types may be authenticated by different tags. A tag shall not be used to authenticate navigation data retrieved after the last Tag-Info field received in the same sub-frame.

Before a tag is used, its ADKD is compared with the one expected from the MAC Lookup Table selected by the DSM-KROOT, and a tag contradicting that sequence is discarded. Where the table defines flexible positions, the ADKD values are assigned dynamically and the corresponding Tag-Info fields are authenticated through MACSEQ, using the same TESLA key and MAC function as the MACK message; this check is not needed when no flexible tags are transmitted. The key used for verification is the one disclosed according to the TESLA schedule, normally in the following MACK message, so that a disclosed key is used to process the tags pending from the previous GST_{SF} ; if that key was not received, it may be derived from a later verified key of the same chain, and in every case it must be verified before use.

The expected tag is then recomputed with the verified key and the same input structure used at transmission time, shown conceptually in Figure 4.7. That input contains the navigation data selected by the ADKD together with the context that fixes the meaning of the tag: PRN_D identifies the satellite whose data are authenticated, PRN_A the satellite associated with the tag, and GST_{SF} binds the computation to the sub-frame of transmission. When $PRN_D \neq PRN_A$ the tag authenticates a satellite other than the one transmitting it, the cross-authentication

case introduced in Section 4.2, which is what allows satellites not broadcasting OSNMA material to be authenticated. The counter CTR identifies the tag position inside the MACK message, preventing a tag from being moved to another position and still accepted, and the NMA status is part of the context as well, so that the computed tag is bound to the service status under which it was generated.

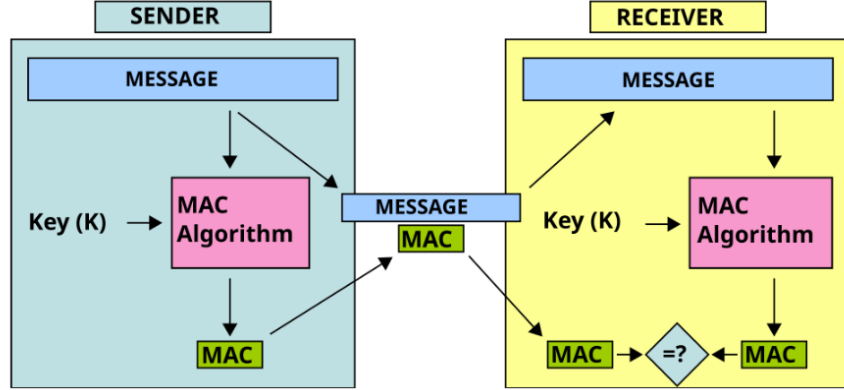


Figure 4.7: Conceptual relation between the received tag, the locally stored navigation data and the verified TESLA key used for tag recomputation[45].

If the computed tag matches the received one, the corresponding navigation data are marked as authenticated. This result applies only to the data selected through the associated ADKD, satellite identifier, issue of data and sub-frame association, and is not a blanket approval of all the data of that satellite.

4.4 Software Architecture and Implementation

The previous sections described OSNMA as a protocol. This one describes the software architecture built in this thesis around it, which combines an OSNMA processing module with a Galileo-only positioning module and exposes the intermediate states of the protocol alongside the receiver observables.

4.4.1 CSSRlib for Galileo-Only SPP

The positioning module requires a solver that computes a navigation solution from raw Galileo observations, so that the authentication status produced by the OSNMA module can be associated with the corresponding position estimates. Rather than implementing one from scratch, this thesis builds on CSSRlib, an open-source Python toolkit for GNSS positioning [21], which already provides the data structures and low-level utilities a basic SPP algorithm needs: satellite

indexing, observation handling, ephemeris management, time representation and coordinate transformations.

CSSRlib was originally designed for high-accuracy processing such as PPP, PPP-RTK and RTK based on SSR or Compact SSR corrections. The objective here is different, namely a Galileo-only Single Point Positioning solution consistent with the observation model of Section 2.7. The implementation therefore reuses the data organisation and several utilities of the library, while removing the elements specific to correction-based positioning, such as precise orbit and clock corrections, tropospheric estimation strategies, ambiguity-related processing and external SSR correction streams. The resulting module estimates the receiver position and clock offset through the Weighted Least Squares procedure described in Section 2.7.1.

The module operates in single-constellation, single-frequency mode on Galileo E1 observations, consistently with the focus of the thesis, since the authentication status provided by OSNMA refers to Galileo navigation data. It can compute a conventional all-Galileo solution, using every available Galileo satellite, or receive an authentication-based satellite selection from the OSNMA module. Running the same algorithm under the two policies is what makes it possible to evaluate the impact of OSNMA on the final PVT solution.

4.4.2 OSNMAlib for OSNMA Processing

An OSNMA-capable system can be realised in different ways: the receiver may implement the whole authentication chain internally, an external module may process the decoded I/NAV and OSNMA stream in real time, or recorded data may be processed offline and inspected together with the positioning solution. This work follows the last approach, which is the one that makes the intermediate states observable: OSNMAlib receives the decoded Galileo navigation data stream, performs the workflow of Section 4.3, and passes its output to the positioning module described above.

OSNMAlib is an open-source Python library for the implementation, testing and analysis of Galileo OSNMA [20]. It processes Galileo I/NAV pages, extracts navigation and authentication data, verifies the required cryptographic material and determines the authentication status of the received information. The I/NAV pages are the result of the receiver demodulating the signal-in-space, as introduced in Section 2.6.3: the receiver tracks the signal, demodulates the navigation bits, groups them according to the page structure and exposes them through a receiver message, either live or recorded in a proprietary binary file. Functionally, the library covers protocol state management, verification of the public key and of the TESLA root key, validation of the chain, tag verification and the association between tags and navigation data, and it supports cold, warm and hot start. In this architecture the cryptographic material is not stored inside the physical receiver

but must be available to the execution environment, so the trust material, that is the Merkle Tree file together with any cached public keys or TESLA parameters, has to be provisioned before execution.

The library accepts several input sources, including Septentrio SBF files, u-blox data, Android GNSSLogger files, GNSS-SDR outputs, live connections and data collected through Galmon. This flexibility is possible because the OSNMA stage only needs the demodulated navigation data and the associated timing information in a suitable format, so the front-end, the receiver model and the logging format can change while the authentication logic stays centred on the same page contents. More recent work by Galan and co-authors also presents OSNMAlib as a platform for improving the Time To First Authenticated Fix, through the recovery of incomplete data, the use of satellite redundancy, the reconstruction of partial messages and dual-frequency reception [46, 47]. Finally, the library provides logging mechanisms, JSON outputs and monitoring tools, which are the basis on which the diagnostic outputs of this thesis are built.

4.4.3 Processing Pipeline

One of the input formats of the pipeline is a recorded UBX file, containing the receiver output of a data collection session. Three message types matter here. The **RXM-SFRBX** messages carry the raw Galileo navigation words, and are therefore the input used to recover the I/NAV pages and the OSNMA fields embedded in the E1-B message; in the authentication branch they are passed to OSNMAlib. The **RXM-RAWX** messages carry the observables measured by the receiver, namely pseudorange, carrier phase, Doppler and carrier-to-noise density ratio; in the positioning branch they feed the SPP module. The **NAV-SAT** messages carry the physical and geometric context used to interpret the results.

RINEX Conversion

For the positioning branch, measurements and navigation data are converted from the proprietary UBX format into RINEX, the Receiver Independent Exchange Format, which separates receiver-dependent logging from the data representation used by positioning algorithms [48]. The observation file provides the code-based measurements required by the SPP algorithm, while the navigation file provides the broadcast ephemeris and clock parameters needed to compute satellite positions and clock corrections. The SPP module uses the two together to associate each measurement with the corresponding satellite position and to solve the Weighted Least Squares problem.

Receiver Time Messages

As discussed in Section 4.3.1, OSNMA processing requires the received pages to be associated with a reliable estimate of GST. A UBX recording may contain time messages such as NAV-TIMEGAL, NAV-TIMEGPS or NAV-TIMEUTC, which report the time estimate computed by the receiver navigation engine, GST_{Rx} , rather than a value decoded from a specific I/NAV word. When available, OSNMAlib can use it to place the raw RXM-SFRBX pages, which carry no absolute time tag of their own, in the correct epoch. This does not replace the content-derived GST_{SIS} used to determine GST_{SF} in Section 4.3.4; it is a complementary reference that eases page association. If the same GST_{Rx} is also used for the consistency check against GST_{SIS} , the independence requirement discussed there still applies.

4.4.4 Diagnostic Outputs

The pipeline produces textual logs, structured files and plots. These outputs exist because the final authentication verdict alone says little: interpreting it requires following how the protocol state evolves, how it interacts with the tracking conditions of Section 2.6.2 and how it affects the SPP solution of Section 2.7.

Two outputs are already available in the baseline OSNMAlib execution. The file `general_logs.log` is a verbose text log recording the main receiver and authentication events, including the first processed GST, the TTFF (Time To First Fix) and the TTFAF (Time To First Authenticated Fix), the latter being the interval between the first processed time reference and the first epoch at which at least four OSNMA-authenticated Galileo satellites are available for an authenticated position solution. The file `status_log.json` is a structured summary storing the OSNMA status at the end of each 30-second sub-frame, including received and authenticated navigation data, verified tags and the public key or chain in force, following the workflow of Section 4.3.4.

Four further outputs were implemented in this thesis. The file `run_metadata.json` stores run-level information, such as whether UBX time messages were used, the first processed TOW, the TTFF, the TTFAF and, when available, the TOW of the first authenticated fix, which provides the TTFAF marker used in the timelines. The file `page_log.csv` contains one row per accepted E1-B I/NAV page, with the satellite identifier, the GST week, the TOW, the sub-frame TOW and the OSNMA flags, and therefore defines the 2-second grid of the extraction timeline; a missing row means that no valid page was processed for that satellite in that interval, because it was absent, filtered out, not part of E1-B, or discarded before entering the processing chain. The file `navsat.csv` stores the satellites reported through NAV-SAT with their carrier-to-noise density ratio, elevation and azimuth, which are not part of the authentication process but provide the physical context needed to interpret it. The file `tag_sources_by_subframe.csv` records, for each

sub-frame, the relation between the satellite whose data are authenticated, PRN_D , and the satellites whose verified tags contributed to that authentication, PRN_A , together with the number of sources and whether self-authentication occurred, so that it is possible to establish who authenticated what according to Section 4.3.4.

The plotting routines combine these files into several views. The satellite extraction state timeline shows, for each satellite and 2-second interval, the state reached while collecting pages, navigation data and OSNMA material, giving a compact view of page availability, complete or incomplete extraction, initialisation, tag acquisition and readiness for later verification; it can also mark the presence of OSNMA bits and the Issue of Data of the received block. The navigation-data authentication timeline works instead at 30-second level and shows whether the data of each satellite became authenticated within a sub-frame, distinguishing absence of authentication, tag-only contribution, cross-authentication and self-authentication, which makes the delayed disclosure mechanism and the cross-authentication relations of Section 4.3.4 directly visible. The colour codes of both timelines are defined in Appendix A.

Alongside these, the pipeline produces the physical-layer and geometry plots: the skyplot of the Galileo satellites reported by the receiver, the C/N0, elevation and azimuth timelines on the same time grid, and the PDOP evolution. They are necessary because authentication events cannot be read from the cryptographic state alone: missing pages, incomplete navigation data or failed authentication may equally follow from weak signal conditions, low elevation, obstructions, multipath or poor geometry.

A further aggregate timeline compares the number of Galileo satellites tracked by the receiver, the number with a complete navigation-data block, the number whose data were actually verified in the sub-frame considered, and the number of missing authentication events, the last counting satellites that had complete navigation data in the previous sub-frame but did not reach an authenticated state in the current one. That curve is a warning indicator rather than an explanation: if most satellites authenticate while one remains in the missing category, the event points to a satellite-specific issue rather than to a global initialisation delay, and the reason has to be investigated through the extraction timeline, the authentication timeline, the tag-source table and the NAV-SAT plots. Overall, the plot gives a compact view of the gap between physical visibility, data extraction and authenticated data availability.

Finally, the positioning branch produces the SPP outputs. The same Galileo-only algorithm runs in *all-Galileo* mode, using every available satellite, and in *authenticated-only* mode, admitting only satellites whose navigation data were authenticated: observables received from a satellite during a 30-second period in which its data were not authenticated are excluded, and if fewer than four authenticated satellites remain, no authenticated-only solution is produced for

that epoch. The resulting positions are plotted as East-North-Up errors against a reference computed with a Canadian PPP service. Comparing the two modes is what makes it possible to evaluate how authentication-based satellite selection affects positioning availability, geometry and accuracy, which is the mitigation-policy question raised in Section 3.3.

All these outputs are stored in a structured form so that they can be reused in the following chapters, both for the comparison of nominal and constrained recordings and as candidate features for the machine-learning dataset built later in this thesis.

Chapter 5

OSNMA Behaviour in Real Reception Scenarios

5.1 Chapter Overview

Chapter 4 described OSNMA as a protocol, and the software pipeline built around OSNMAlib and CSSRlib. This chapter uses that pipeline on real recordings. The purpose is to verify that the implemented outputs actually make the protocol observable and interpretable, and to establish what those outputs resolve that the receiver observables alone would leave ambiguous.

Two static recordings are used as reference: a *nominal* scenario, with clear sky view, and a *constrained* scenario, with roughly half of the sky blocked by a building. The nominal recording provides the baseline against which every anomaly of the constrained recording is read. The chapter then moves from the aggregated view to the full diagnostic output, from there to single-satellite events, and finally to the effect of authentication on the position solution.

5.2 Experimental Setup

Both recordings were collected at DLR Neustrelitz with a u-blox NEO-M8T receiver configured for Galileo-only, single-frequency (E1) operation, and logged as raw UBX. Their parameters are summarised in Table 5.1, and the nominal installation is shown in Figure 5.1: the antenna is visible on the roof edge in the photograph, and its position on the site map is marked on the right.

The whole cryptographic chain is executed by OSNMAlib; the physical-layer context (C/N_0 , elevation, azimuth, PDOP) is parsed directly from the raw UBX NAV-SAT and NAV-DOP messages, so that authentication events and signal conditions

Table 5.1: Reference recordings used in this chapter.

	Nominal	Constrained
Receiver	NEO-M8T	NEO-M8T + ANN-MB antenna
Signal	Galileo E1, single freq.	Galileo E1, single freq.
Date	29 April 2026	30 April 2026
Interval	13:53–13:58	07:05–07:20
Duration	5 min	15 min
Antenna	rooftop, unobstructed LOS	first-floor window ledge
Environment	clear sky view	~180° sky block, multipath

share one time axis. Time synchronisation for the offline runs is taken out through the host machine, as discussed in Section 4.4.3.



Figure 5.1: Nominal static setup: rooftop antenna and its position on the site.

5.3 Aggregated View of the Authentication Process

The first diagnostic output is an aggregated, per-subframe count of four quantities, introduced in Section 4.4.4:

- **tracked satellites:** Galileo satellites reported by the receiver;
- **complete navigation data:** satellites for which a full ADKD 0 block was assembled in the current subframe, awaiting a TESLA key or KROOT before it can change state;
- **authenticated navigation data:** satellites that completed the authentication process in the current subframe, by self- or cross-authentication, including data stored in earlier subframes and validated only now, which makes the memory mechanism of the protocol visible;
- **missed authentication:** satellites that had complete navigation data in the previous subframe and did not reach an authenticated state in the current one.

5.3.1 Nominal Scenario

Figure 5.2 reports those four counts for the nominal recording. The three curves converge immediately and stay flat: ten satellites deliver complete navigation data from the first subframes (orange), all of them become authenticated once the KROOT is assembled (blue), and the missed-authentication count (red) remains at zero for the whole recording while the tracked count (green) grows from twelve to thirteen. The delay between the orange and the blue curve is the initialisation interval discussed in Section 5.5.1. The nominal case therefore provides an unambiguous reference: full extraction, full verification, no gaps.

5.3.2 Constrained Scenario

The same plot for the constrained recording, in Figure 5.3, behaves differently. Only about half of the tracked constellation reaches complete navigation data, and the missed-authentication curve becomes non-zero at several epochs, visible as the red peaks around 400 s and after 750 s. Those peaks are the useful part: during the same subframes, the rest of the constellation authenticates normally, so the event specific to some satellites. This is what makes the aggregated plot a preliminary warning indicator only. It states that something happened to specific satellites; it does not state whether the cause is multipath, local interference or an attack. The

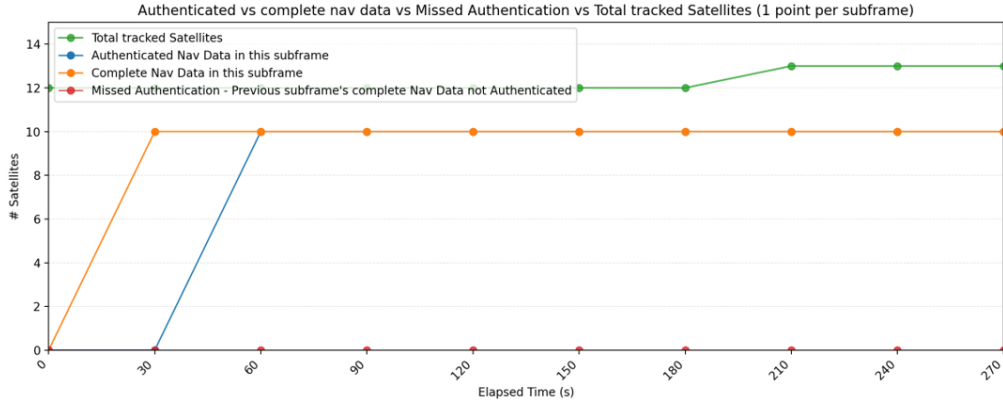


Figure 5.2: Nominal scenario: tracked, complete, authenticated and missed authentication counts, one point per 30-second subframe.

root cause must be looked for at the single-channel level, which is the subject of Section 5.5.

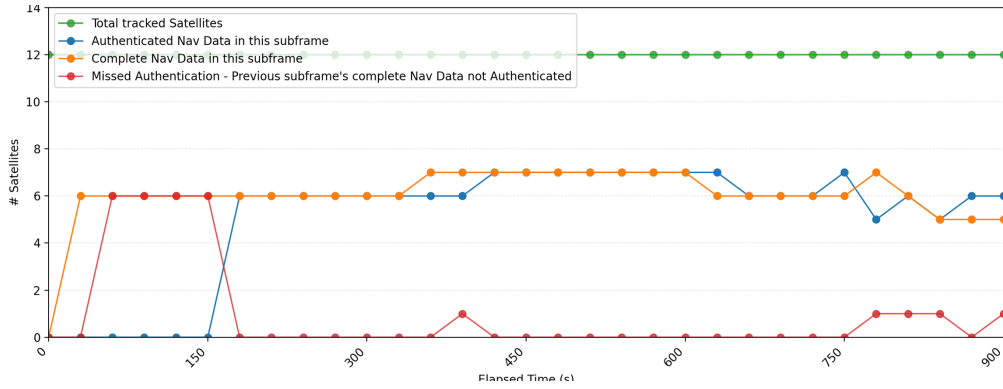


Figure 5.3: Constrained scenario: the same four counts. The non-zero red peaks mark satellites that had complete navigation data in the previous subframe and did not reach an authenticated state in the current one.

5.3.3 Signal Conditions Behind the Difference

The skyplot of the constrained recording, in Figure 5.4, explains the asymmetry directly, and the corresponding C/N_0 timeline is reported in Figure 5.5. Satellites in the northern half of the sky, facing the window where the antenna was placed, show strong and stable signals around 35–40 dB-Hz, which appear as the green rows of the timeline, while satellites in the southern half stay between 0 and 20 dB-Hz for the whole recording, appearing as darker rows. The window provides a clear

line of sight in the facing azimuth, and the building physically blocks everything behind it. Extraction failures, incomplete subframes and missed authentications concentrate on the blocked half.

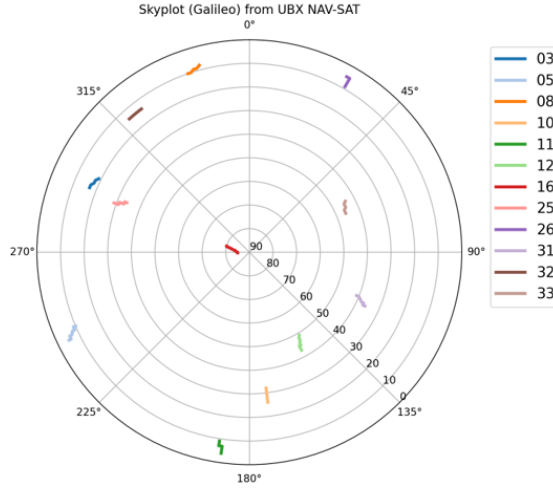


Figure 5.4: Constrained scenario: skyplot of the Galileo satellites reported by the receiver, built from UBX NAV-SAT.

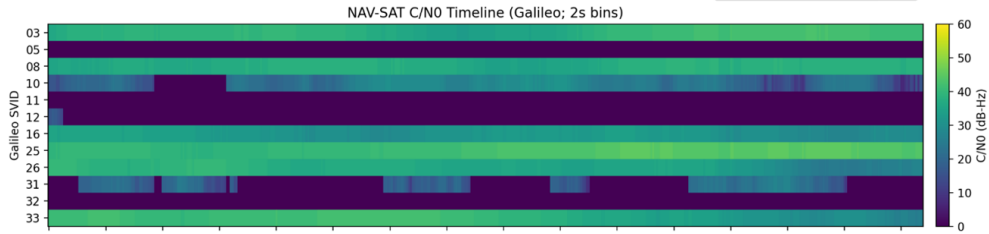


Figure 5.5: Constrained scenario: C/N_0 timeline per satellite, showing the north/south asymmetry produced by the building.

5.4 The Full Diagnostic Output

Before isolating individual events, it is useful to look at the complete output of a single run, since every event discussed in the next section is a portion of it. Figure 5.7 shows the constrained recording as the pipeline produces it, with all panels sharing the same time axis. From top to bottom:

1. the **satellite extraction state timeline**, in 2-second bins, which follows the receiver's progress in collecting pages, assembling navigation data and acquiring MACK tags, with a violet outline marking the subframes in which OSNMA bits are present;
2. the **navigation-data authentication timeline**, in 30-second subframe bins, which reports the outcome of the verification, distinguishing no authentication, tag-only contribution, cross-authentication and self-authentication, with an orange outline marking the satellites acting as cross-authenticators;
3. the **C/N₀ timeline**, which gives the signal quality behind each of the states above;
4. the **elevation and azimuth timelines**, which give the geometry of each satellite over the same interval;
5. the **PDOP timeline**, which summarises the geometry of the constellation as a whole and is the quantity referenced in Section 5.6.

Read vertically, the figure associates a cryptographic state with some of the physical conditions under which it occurred, which is the property that makes the events of the next section interpretable at all, both by human and a machine learning model: without the physical conditions, a satellite that stops authenticating is indistinguishable from a satellite that stops being received. Figure 5.6 reports the same output for the nominal recording, where the two upper timelines are uniform and the C/N₀ panel shows no dark rows, which is the comparison the constrained case is read against.

Each of the event figures in Section 5.5 is a zoom of the two upper timelines of Figure 5.7, highlighted by the boxes drawn on top of the figure, restricted to a few satellites and a few subframes, and reproduced at the same scale so that they can be compared with one another. The red, yellow and black annotations in those zooms mark the cells discussed in the text. Those zooms will be presented from left to right looking at boxes.

5.5 Single-Satellite Events

The extraction-state timeline and the navigation-data authentication timeline, whose state definitions are reported in Appendix A, allow individual events to be isolated. The cases below are the ones that happened to be in the constrained environment recording shown earlier and that a detection layer would have to distinguish from an attack. However those are most likely not real attacks because the recording was still collected in a controlled environment with no known threat being exploited. Those are possible interpretations of events.

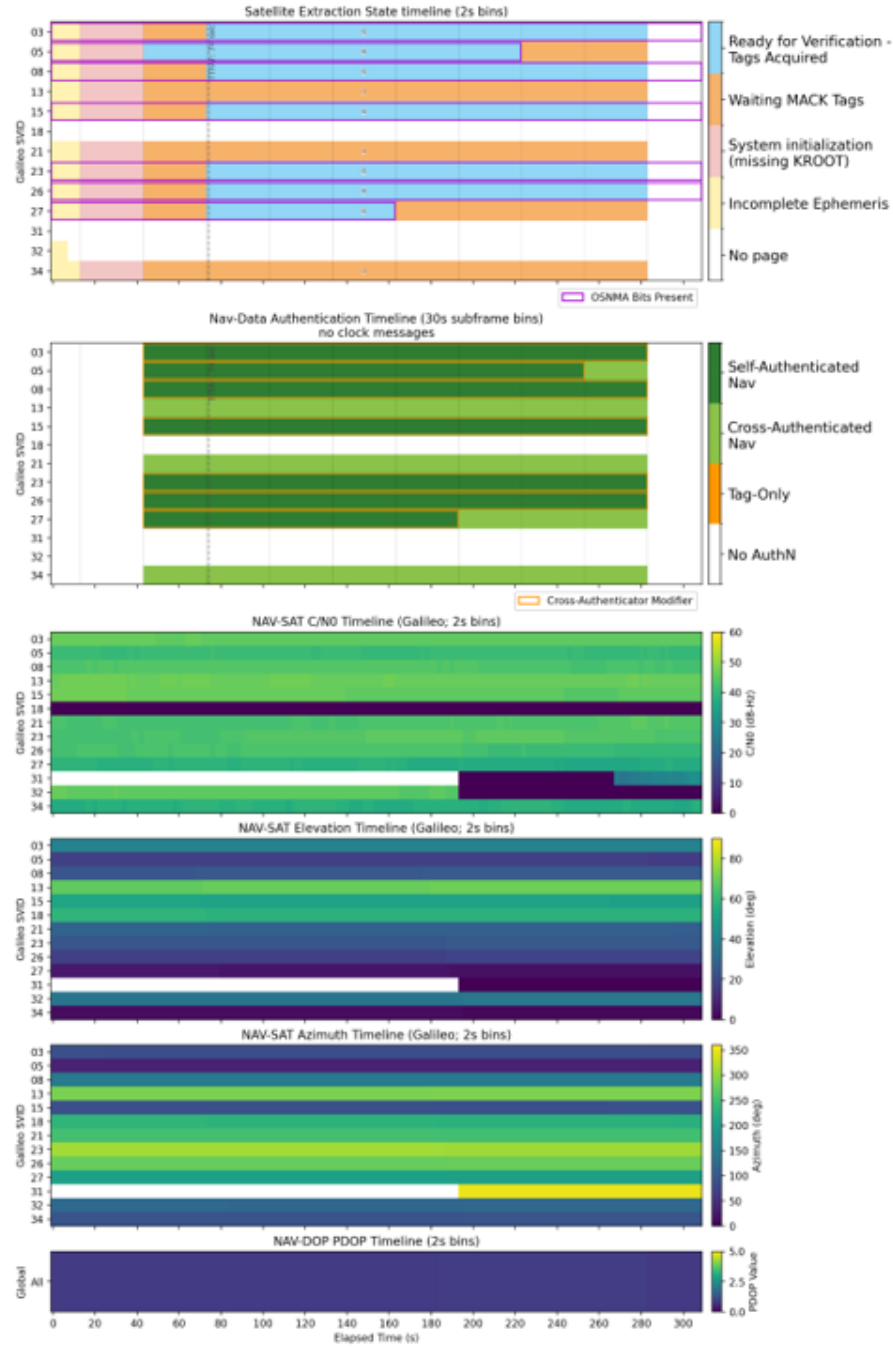


Figure 5.6: Nominal scenario, full diagnostic output, shown for comparison with Figure 5.7.

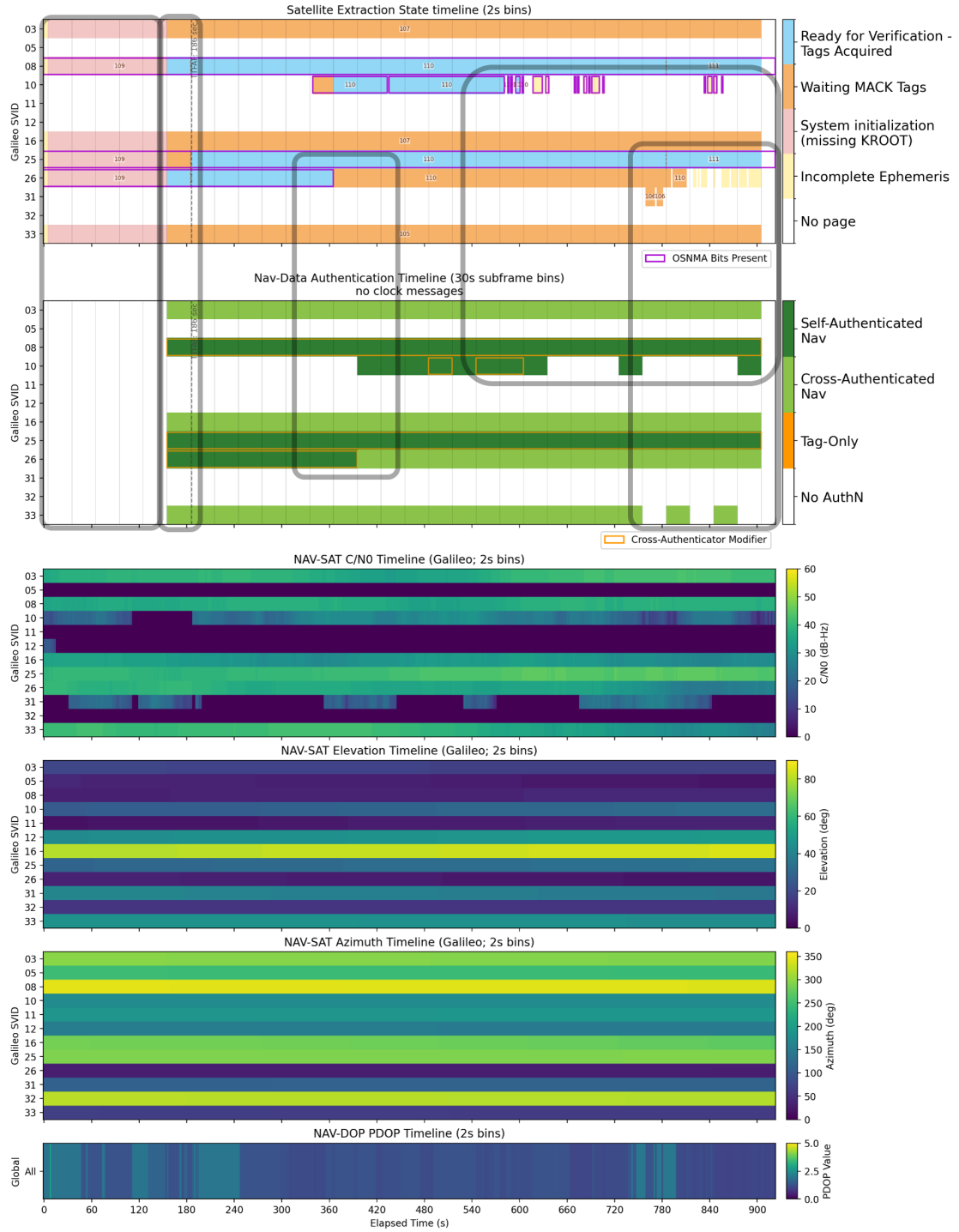


Figure 5.7: Constrained scenario, full diagnostic output: extraction state and authentication timelines on top, C/N_0 , elevation, azimuth and PDOP below, all on a common time axis.

5.5.1 Initialisation and the TTFAF Boundary

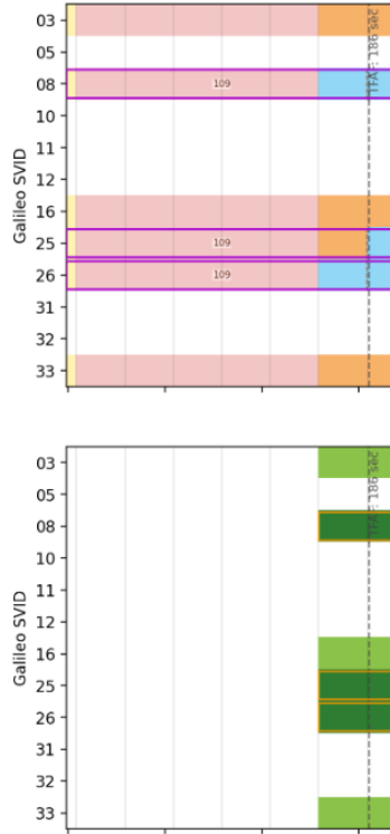


Figure 5.8: Initialisation and the TTFAF boundary: extraction state (top) and authentication timeline (bottom) around the first authenticated subframe.

Complete ephemerides are extracted almost immediately, within the first subframe. Authentication, however, is blocked: the receiver stays in a holding state while the DSM-KROOT blocks accumulate, even though the satellites are already broadcasting OSNMA bits. During this interval the navigation data are accumulated but strictly unverified, since no root of trust has been established yet. Both states are visible in Figure 5.8: the pink cells of the upper timeline are the satellites waiting for the KROOT, and the corresponding cells of the lower timeline stay white, even for the satellites whose OSNMA bits are already being received.

This also explains where the TTFAF marker, the dashed vertical line in Figure 5.8, falls. The KROOT is not transmitted in a single burst: its DSM blocks are spread over several 30-second subframes, so the receiver must wait for the last page of the last block before the key can be assembled. Only at that final second does

the subframe become authenticated, and the first-authenticated-fix timestamp is therefore recorded at the subframe boundary immediately following the extraction, not at its beginning.

5.5.2 TESLA Delay and the Cross-Authenticator Role



Figure 5.9: TESLA delay and the cross-authenticator role. Red: self-authentication with no tags extracted in the current subframe. Yellow: cross-authentication of a satellite broadcasting no OSNMA bits.

A recurring combination, circled in red in Figure 5.9, is a satellite whose extraction state shows no tags acquired in the current subframe while its navigation data are nevertheless self-authenticated. This is the delayed-disclosure mechanism working as designed: the TESLA key disclosed in the current subframe validates a tag received in the previous one. In the same event the satellite frequently carries the cross-authenticator marker, the orange outline of the lower timeline, meaning that its own past tags are being used, in the same subframe, to authenticate the

navigation data of other satellites.

The complementary case, circled in yellow in the same figure, is a satellite that nominally broadcasts zero OSNMA bits by design, recognisable by the absence of the violet outline in the upper timeline. It cannot self-authenticate, since it provides no cryptographic material of its own, and its navigation data are verified through tags broadcast by other satellites in the previous subframe, which is why it appears in the lighter cross-authenticated shade. Both cases are nominal and produce, at the aggregate level, exactly the counts that a naive reading would classify as anomalous.

5.5.3 Satellites Never Cross-Authenticated

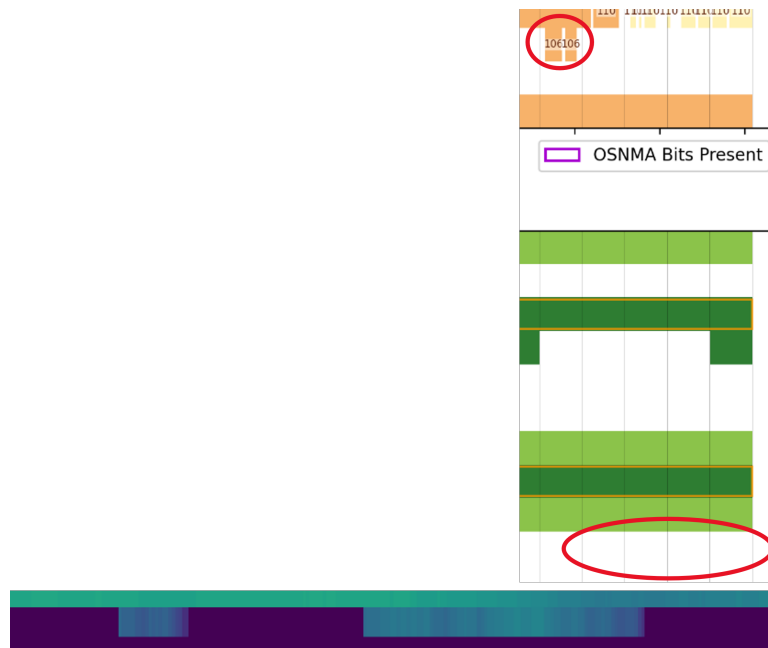


Figure 5.10: A satellite never cross-authenticated: scattered page extraction, an empty authentication row, and the C/N_0 trace explaining both.

A satellite in the blocked half of the sky (SV 31) delivers only a few scattered pages over the entire recording, circled in red in the upper timeline of Figure 5.10. Its complete navigation data are eventually accumulated, but no OSNMA tag is ever retrieved from its own signal, and its authentication timeline remains empty for the whole recording, the blank row circled in the lower panel: no other satellite ever broadcasts a MACK tag targeting its navigation data, although this would be possible in principle. The C/N_0 timeline at the bottom of the figure shows extremely poor signal quality outside the few subframes where pages were received,

which indicates that the cause is poor line of sight rather than a nominal absence of OSNMA material from that satellite.

5.5.4 The Memory Mechanism Under Signal Degradation

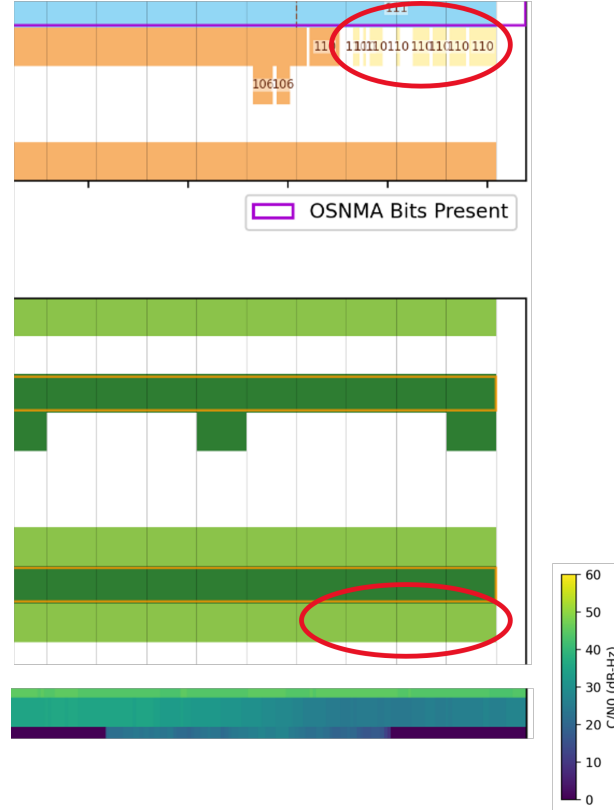


Figure 5.11: The memory mechanism: page losses in the extraction timeline with an uninterrupted cross-authenticated state below.

SV 26 shows the opposite outcome under a comparable degradation, reported in Figure 5.11. Its C/N_0 falls from roughly 40 to 20 dB-Hz, the receiver starts dropping pages, visible as the yellow cells of the upper timeline, and the reception of new complete navigation data halts. Its authentication timeline nevertheless remains uninterrupted in the cross-authenticated state, the continuous row circled in the lower panel. Two mechanisms are responsible: the receiver retains the last fully extracted navigation block, and it buffers individual ADKD 0 words, so that fragments decoded across several subframes can still be assembled into a complete set. Meanwhile other satellites keep broadcasting MACK tags that validate the stored data. A satellite can therefore remain a cryptographically trusted source while its own data stream is temporarily compromised.

5.5.5 Abrupt Zeroing of the OSNMA Field

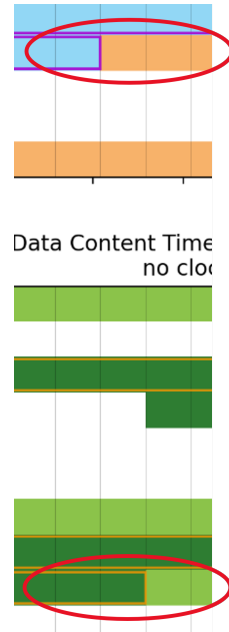


Figure 5.12: Abrupt zeroing of the OSNMA field: the violet outline disappears while extraction and signal quality remain nominal.

A qualitatively different event was observed on the same satellite, and is circled in Figure 5.12. C/N_0 stays strong and stable, complete ephemerides are extracted, the subframes are structurally sound, and yet the OSNMA field drops to exactly zero: the violet outline marking the presence of OSNMA bits disappears from one subframe to the next. The navigation data remain intact and the receiver processes the zeros without an integrity failure, so the satellite simply becomes cross-authenticated by the rest of the constellation, which is the change of shade circled in the lower timeline.

The signature is worth flagging because only the cryptographic field disappears while everything else stays flawless. Two readings are possible: a commanded, temporary suspension of the OSNMA broadcast by the ground segment, or a targeted downgrade attempt in which an attacker forwards the legitimate navigation data and zeroes the authentication field, so that no signal-loss or high-noise alarm is triggered. The recording alone cannot separate the two, which is precisely the point: the event is indistinguishable at the physical layer and visible only in the OSNMA state.

5.5.6 Progressive Loss of Authentication

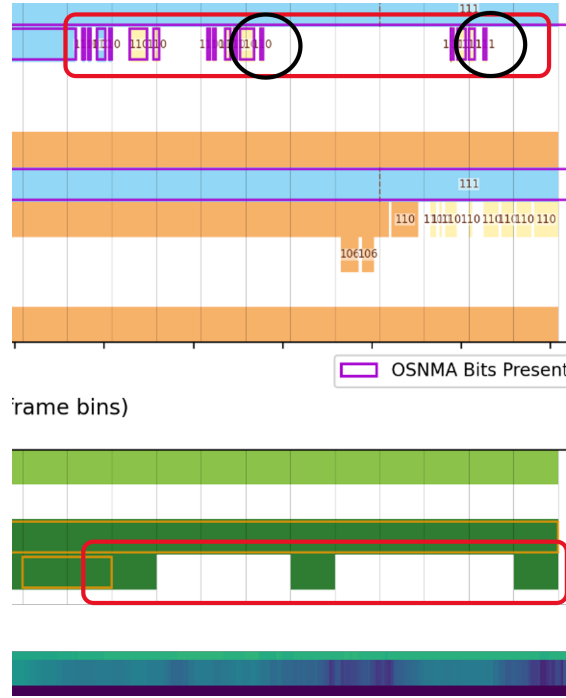


Figure 5.13: Progressive loss of authentication on SV 10, with the incomplete-extraction states marked in black and the unverified stretch marked in red.

SV 10 degrades gradually, as shown in Figure 5.13. It first loses its cross-authenticator role while still retrieving a self-authenticating tag, that is, the orange outline disappears while the cell stays in the self-authenticated shade, then stops extracting complete navigation data altogether and remains unverified for three consecutive subframes, the blank stretch circled in the lower timeline, since no other satellite broadcasts cross-authenticating tags for its stored data. The intermediate state circled in black, in which single-subframe extraction is incomplete because of page losses while OSNMA bits are still present and a new MACK tag is parsed, shows the recovery path: the fresh tag self-authenticates data assembled from older fragments. Intermittent signal drops are therefore survivable, provided at least one usable tag arrives.

5.6 Impact of Authentication on the Position Solution

The final step translates authentication into positioning. The same Galileo-only SPP solver was run in the two modes introduced in Section 4.4.1, `all_galileo` and `auth_only`, against a reference position computed with the Canadian PPP service giving as input the same recording used to display those figures.

5.6.1 Nominal Scenario

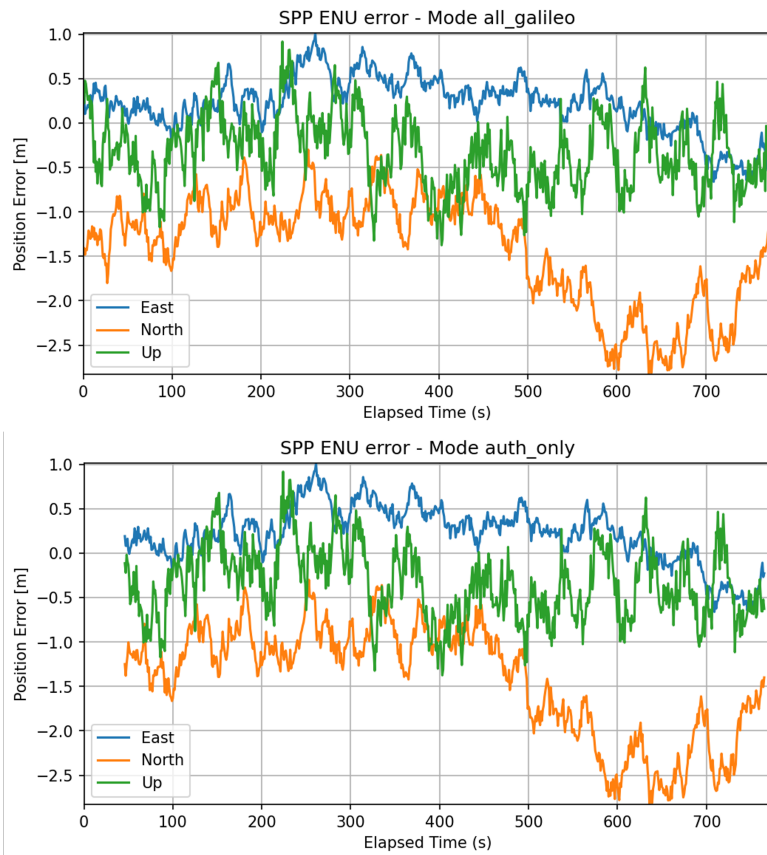


Figure 5.14: Nominal scenario: `all_galileo` (top) against `auth_only` (bottom) SPP ENU error.

In nominal conditions the all-Galileo solution is stable, with an ENU error below three metres over the whole recording, as shown in the upper plot of Figure 5.14. The `auth_only` solution, in the lower plot, starts later, since no position is produced before four satellites are authenticated, and from that point on it overlaps the

all-Galileo trace exactly. The overlap confirms that every tracked satellite keeps its authenticated status without dropping, so having authentication costs nothing in accuracy in this particular case.

5.6.2 Constrained Scenario

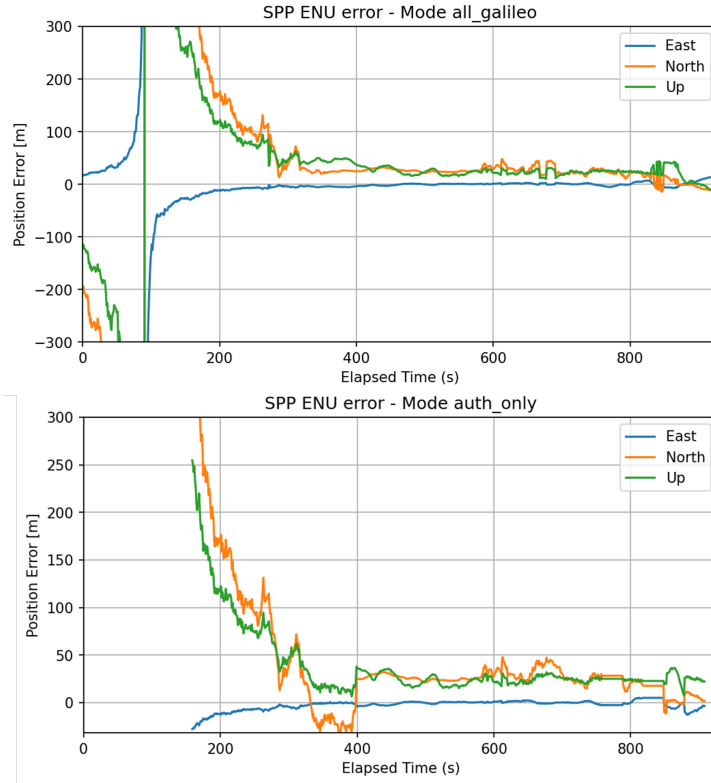


Figure 5.15: Constrained scenario: `all_galileo` (top) against `auth_only` (bottom) SPP ENU error. The two traces no longer overlap.

In the constrained recording the all-Galileo solution shows error spikes above 300 m before collapsing back into the tens of metres, in the upper plot of Figure 5.15. The degradation does not correlate with an overall C/N_0 deterioration in that interval; it follows a severely degraded PDOP, which improves as the geometry recovers, and which can be read on the bottom panel of Figure 5.7. This also raised a practical requirement for data collection: a receiver computing PDOP over all visible constellations does not describe the geometry actually used by a Galileo-only solution, so the receiver must be strictly configured to track Galileo only.

The two modes no longer overlap. The `auth_only` trace starts only after the first authenticated fix and follows a distinct error profile, because it relies on a

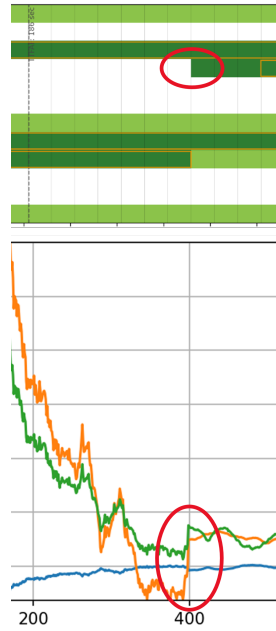


Figure 5.16: Constrained scenario: the step in the `auth_only` error (left) produced when SV 10 completes its authentication, absent from the `all_galileo` error (right).

restricted and continuously changing subset of trusted satellites. One discontinuity is directly traceable to the authentication timeline, and is shown in Figure 5.16: at the epoch where SV 10 completes its authentication process, circled in the authentication timeline on top, the authenticated-only trajectory on the left shows a sharp step that is completely absent from the all-Galileo trace on the right. The step is produced by the sudden addition of new observables to the estimator, and is an example of authentication state propagating into the position domain.

5.6.3 Retrospective Selection

A third mode was also produced, in which the observables retained are those belonging to subframes whose navigation data were *eventually* authenticated at some later point in the timeline. Its error trace closely mirrors the standard all-Galileo one, because under these observation conditions all stored navigation data passed OSNMA verification sooner or later. The mode is reported for completeness only but not developed here: it uses information carried by future MACK tags and TESLA keys, so it can exist exclusively in post-processing.

5.7 Summary

The implemented outputs make the OSNMA state observable end to end, and the single-satellite events of Section 5.5 demonstrate that observability can be worth. Read from the aggregated counts alone, each of those events looks anomalous: a satellite that authenticates without having acquired any tag in the current subframe, one whose authentication row stays empty for the whole recording, one that holds a cryptographically trusted state while its own data stream breaks down. Read in the full output, each resolves into an identifiable protocol state, attributable to a specific satellite and a specific subframe: delayed key disclosure, cross-authentication, data buffering. The resolution is possible because the timelines carry the cryptographic state and the physical conditions on one axis; without the lower panels, a satellite that stops authenticating cannot be told from a satellite that stops being received.

That structure is what this chapter contributes to the rest of the work. Distinctions of this kind are not reachable from the receiver observables alone, and they take the form of per-satellite, per-subframe patterns, which is the form a detection layer can learn from. This is what makes OSNMA-derived quantities a candidate as an additional input to the models later on. They are deliberately left out of the next data-driven chapters, which operate on physical-layer observables only, so that their contribution can be measured against a baseline that does not already contain them.

The two recordings also delimit what each layer is able to see. The events that are genuinely suspicious, such as the selective zeroing of the authentication field, are invisible to physical-layer monitoring, while the events that damage positioning the most in these recordings, such as degraded geometry, are invisible to OSNMA. Neither layer is sufficient alone. This motivates the data-driven approach developed from Chapter 8 on.

Chapter 6

Live Authentication and Start-Mode Analysis

6.1 Chapter Overview

This chapter moves to a second receiver generation, the u-blox ZED-X20P. It first measures the Time To First Authenticated Fix across start modes, and examines in detail two study cases in which start modes that should differ return the same value. It then turns to recordings in which the received signal is no longer trustworthy: a controlled spoofing scenario generated through an SDR, whose ground truth is known by construction, and a portion of a Jammertest 2025 recording spanning the boundary between a nominal interval and a declared meaconing and spoofing transmission.

6.2 Start Modes and Time to First Authenticated Fix

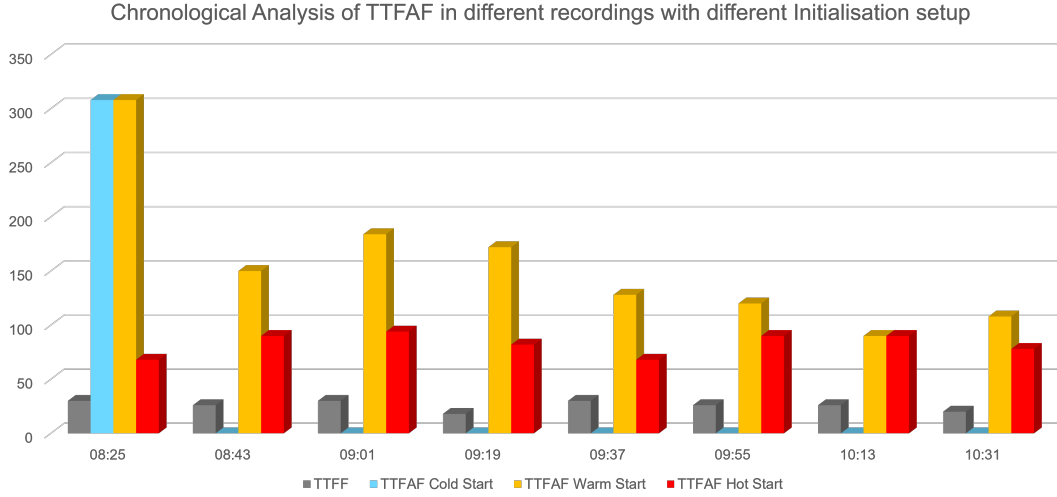
6.2.1 Nominal Setup

Eight consecutive static recordings of fifteen minutes each were collected with a ZED-X20P on 16 July 2026, between 06:25 and 08:25, in Galileo-only single-frequency configuration, with the antenna on the roof and unobstructed line of sight. Each recording was processed three times, forcing a cold, warm and hot start by controlling the cryptographic material available to the software at run time.

Under these conditions the `all_galileo` and `auth_only` solutions are identical, consistently with Section 5.6: with clear sky view, enforcing authentication does not remove any satellite from the solution.

Table 6.1: Average TTFAF over the eight rooftop recordings, by start mode.

Start mode	Average TTFAF [s]
Cold	not reached within the recording
Warm	157
Hot	82

**Figure 6.1:** TTFAF per recording and start mode over the eight rooftop sessions.

6.2.2 Cold and Warm Starts having Same TTFAF

In one recording the cold-start and warm-start runs returned the same TTFAF. The OSNMAlib logs show why: a subframe carried a DSM-PKR inside the HKROOT message, the receiver reconstructed and verified the public key against the Merkle tree, and the run transitioned from cold to warm start. While the DSM-PKR blocks occupy the HKROOT stream, the DSM-KROOT cannot be transmitted, so the parallel warm-start run, which already had a trusted public key and was only waiting for the KROOT, was delayed by exactly the same amount. The two runs therefore converge, and the equivalence comes from a the HKROOT channel being shared between the two DSM types.

6.2.3 Warm and Hot Starts Collapsing, and a Proposed Change

A second equivalence, between warm and hot start, was traced to the way applicable tags are identified. In the first subframe of the hot-start run, MACK tags were received and complete navigation data were extracted for several satellites, but

nothing could be verified yet, since the disclosing key had not arrived. In the second subframe the key arrived and the MACSEQ of the previous subframe could be checked, but the tags of the first subframe found no navigation data to apply to. The cause is an implementation choice: OSNMAlib admits only navigation data received in the subframe strictly preceding the one from which the tag was extracted. Authentication therefore happened only one subframe later, when the tags of the second subframe were verified against data collected in that same interval.

The Receiver Guidelines only forbid using a tag on navigation data retrieved after the last Tag-Info field of the subframe in which the tag was transmitted; they do not forbid using data retrieved earlier within that same subframe. Allowing tags to apply to navigation data extracted in their own subframe is therefore compliant, and would remove the collapse described above. This is proposed as a modification to OSNMAlib.

6.3 Constrained Scenario with the ZED-X20P

The constrained geometry of Section 5.2 was repeated with the ZED-X20P and a low-cost ANN-MB antenna, in a single 1 h 15 min recording on 15 July 2026, between 14:30 and 15:45, from the same first-floor window ledge. Starting in warm mode, the run reached a TTFF of 18 s and a TTFAF of 202 s.

The diagnostic outputs reproduce the picture of Chapter 5 on a different receiver: the skyplot shows the same directional asymmetry, the aggregated timeline shows the gap between tracked, complete and authenticated satellites, and the two SPP modes diverge, since the authenticated subset is smaller and changes over time. Obtaining the same qualitative behaviour with a different receiver generation is what allows those observations to be attributed to the reception geometry and to the protocol rather than to one specific piece of hardware.

6.4 SDR-Based Spoofing Simulation

A controlled spoofing scenario was finally set up to obtain a recording in which the ground truth is known by construction. A digital signal is generated and streamed through GNU Radio to an SDR, which, synchronised by an external rubidium clock, transmits the corresponding analogue RF signal to the receiver; the ZED-X20P processes it and outputs UBX messages to u-center 2. The recording lasts six minutes, on 3 August 2026 between 13:51 and 13:56, in GPS-only single-frequency (L1) configuration.

The same diagnostic outputs were produced: skyplot from the received telemetry, C/N₀ timeline, the number of satellites known to be visible or tracked against the

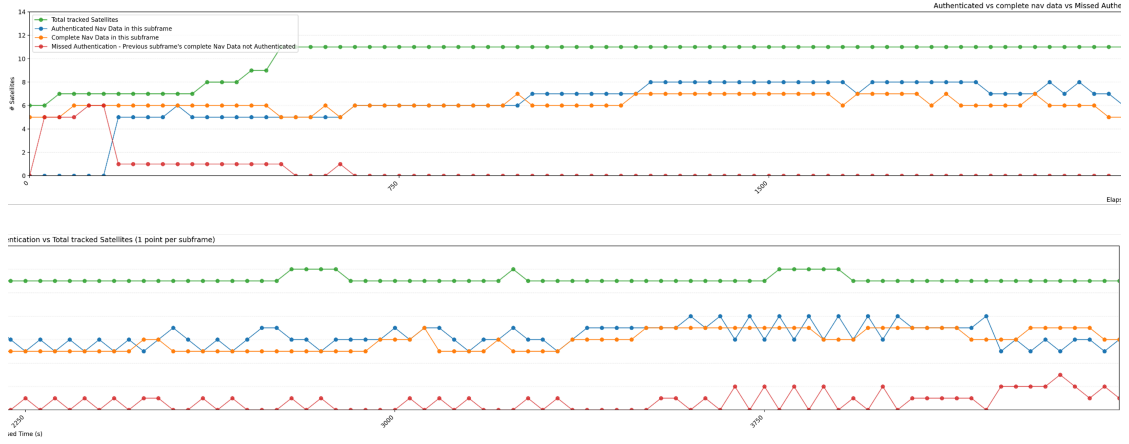


Figure 6.2: Constrained scenario with the ZED-X20P: aggregated timeline of tracked, complete and authenticated satellites.

number actually used in the position estimation, and the SPP ENU error. The scenario is GPS-only, so it carries no OSNMA content; its role is to provide a reference signature of a counterfeit constellation in the observable domain, which is the domain the models of the following chapters operate on.

6.5 A Jammertest Recording With Declared Meaconing and Spoofing

The SDR scenario of Section 6.4 provides a spoofing signature with known ground truth, but the receiver was fed by cable and never received an over-the-air signal. This section closes that gap with a thirty-minute recording, 17:30–18:00, of a live Septentrio SBF recording made during the Jammertest 2025 campaign, the one that supplies the labelled dataset of Chapter 8 (Section 8.2), at the same site, on Galileo E1, single frequency, on the 16th of September 2025. The recording follows the declared transmission boundary: the first fifteen minutes, 17:30–17:45, carry no declared transmission, while from 17:45 meaconing and spoofing were declared on this site according to the campaign’s official transmission plan, the same plan that provides the ground truth of Section 8.2. Unlike Section 6.4, the receiver here was tracking an over-the-air signal at an operational test range, so this is the closest this thesis comes to a genuine attack.

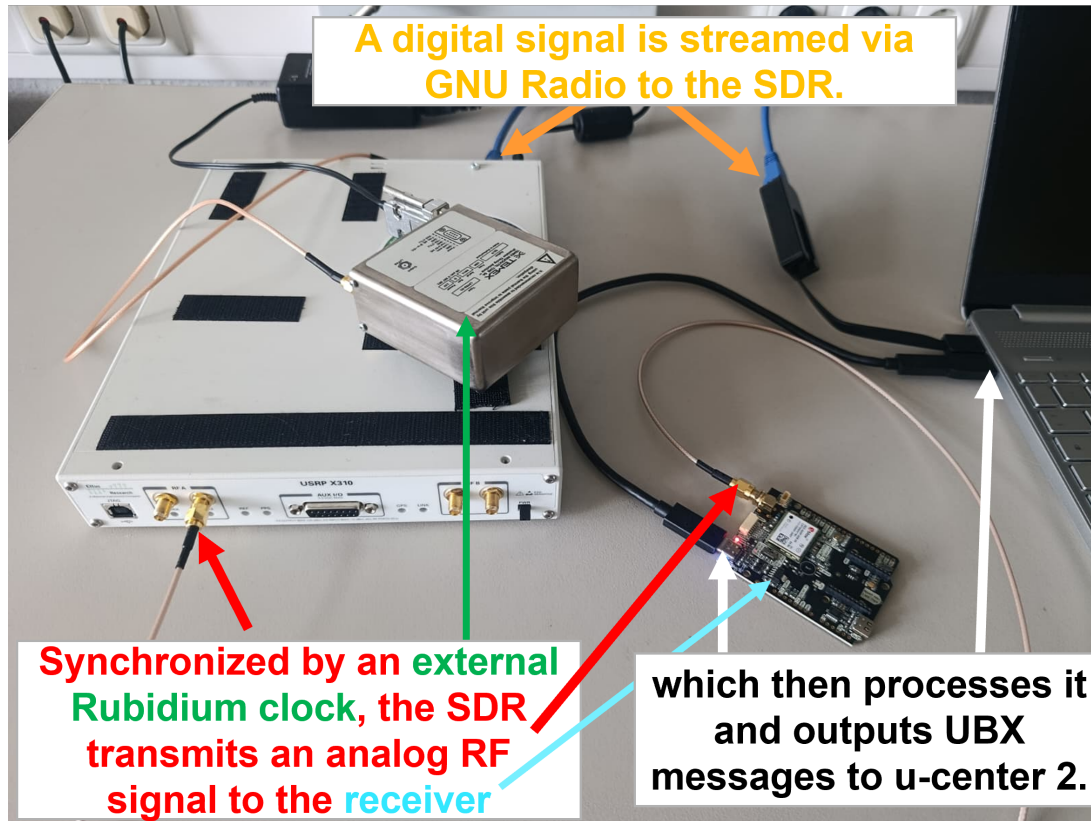


Figure 6.3: SDR spoofing setup: GNU Radio, rubidium-disciplined SDR and receiver.

Table 6.2: Reference recording used in this section.

Receiver	Septentrio
Signal	Galileo E1, single freq.
Date	16 Sep 2025
Portion shown	17:30–18:00
Nominal segment	17:30–17:45
Declared attack	17:45–18:00, meaconing and spoofing
Ground truth	Jammertest 2025 transmission plan

6.5.1 Aggregated View

Figure 6.8 reports the same four per-subframe counts introduced in Chapter 5: tracked, complete, authenticated and missed-authentication satellites, over the 17:30–18:00 portion. The first half, 17:30–17:45, reproduces the nominal pattern

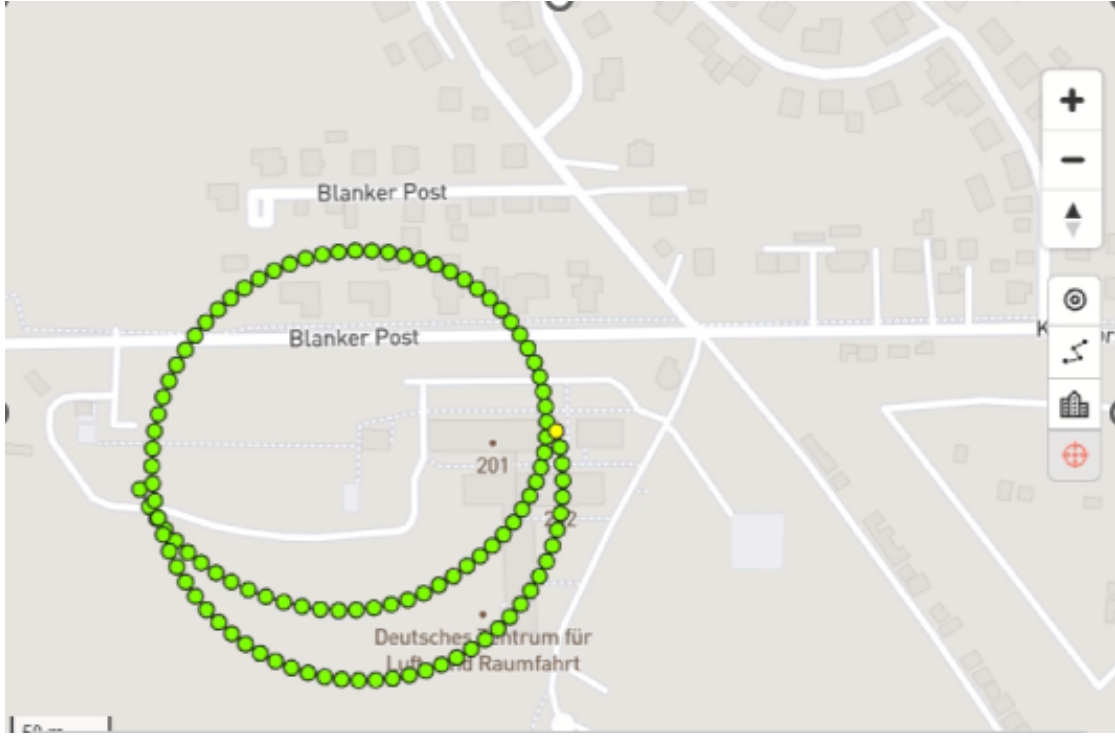


Figure 6.4: Spoofed circular trajectory visualized in u-center 2.

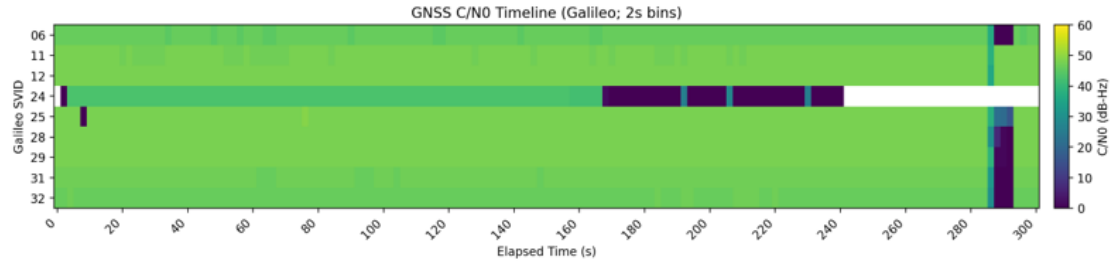


Figure 6.5: C/N_0 timeline of the SDR spoofing scenario.

of the clear-sky recording of Chapter 5: the curves converge and the missed-authentication count stays most of the time at zero. Past 17:45 the picture changes:

after a brief transient in which the tracked-satellite count returns to its nominal level, missed authentication reaches its maximum and authenticated data drops to zero for the remainder of the recording. Satellites continue to be tracked and to deliver complete navigation data, but none of it is authenticated again. This is the one recording in the thesis where the failure is total and constant throughout the whole attack.

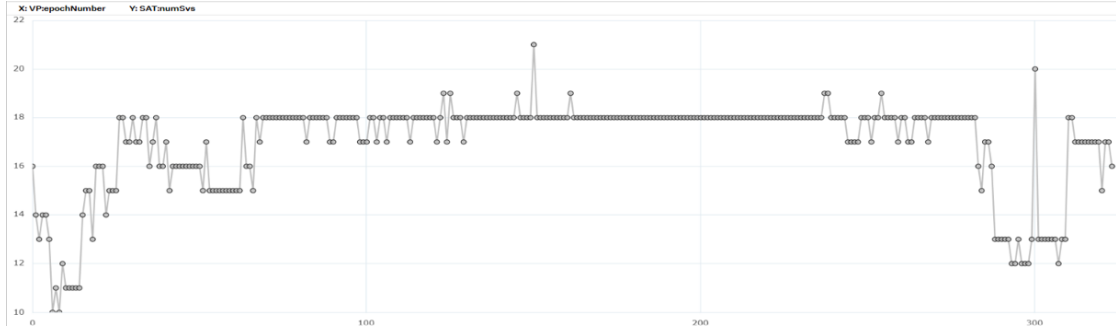


Figure 6.6: Number of satellites tracked by the receiver in that epoch.

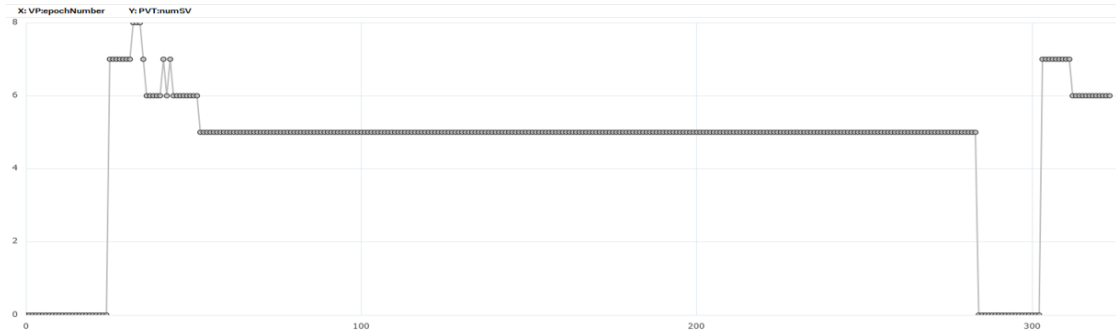


Figure 6.7: Number of satellites used in position estimation in that epoch.

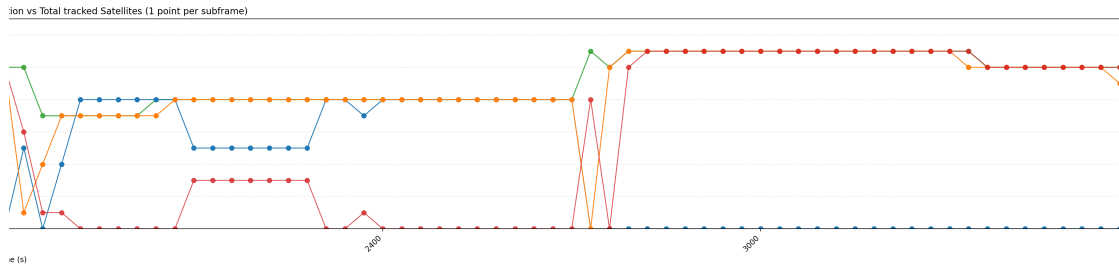


Figure 6.8: Tracked, complete, authenticated and missed-authentication counts over the 17:30–18:00 period. The nominal segment (17:30–17:45) matches the clear-sky case of Chapter 5; the declared attack begins at 17:45 and ends in a sustained loss of authentication that persists to the end of the recording.

6.5.2 Skyplot and Reception Conditions

Figure 6.9 shows the skyplot for the recording. Reception conditions were unobstructed, which takes out a geometric explanation for the authentication loss described above.

A subset of the tracked satellites (SVs 11, 12, 18) appears only after the declared

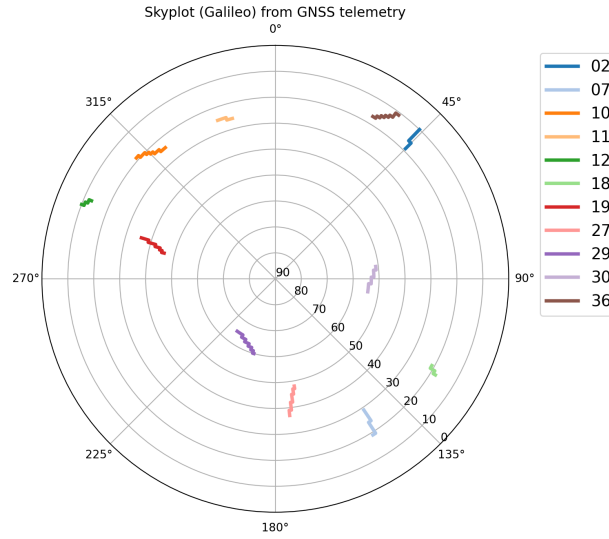


Figure 6.9: Skyplot of the Galileo satellites tracked during the recording.

attack begins, absent from the first fifteen minutes of the recording and present at a strong signal level once the attack window starts. This can be seen in Figure 6.9, since those 3 satellites have a shorter tracked path compared to the other satellites, satellites appearing out of nowhere in numbers or at signal strengths that even the real, unobstructed sky at this site does not justify. Their elevation makes the point sharper. All three sit very low, and SV 36, which is tracked throughout at a comparable elevation, holds a visibly lower C/N_0 for the whole recording, as the last row of the C/N_0 timeline of Figure 6.10 shows. Three satellites at that elevation should look like SV 36; instead they arrive strong and stay strong.

6.5.3 Full Diagnostic Timeline

Figure 6.10 reports the full diagnostic output introduced in Chapter 5 for the whole recording: per-satellite extraction state, navigation-data authentication timeline, and C/N_0 . Three moments are worth isolating.

In the first half the authentication timeline is populated: satellites hold self- and cross-authenticated states subframe after subframe. At the boundary the timeline empties in one step and stays empty, with not a single authentication for the rest of the excerpt, while the extraction state stays in *Waiting MACK Tags* across the constellation. The C/N_0 panel moves in the opposite direction over the same boundary: levels rise slightly for every satellite rather than falling. And the three satellites of Section 6.5.2, absent from the first half, appear here at the boundary

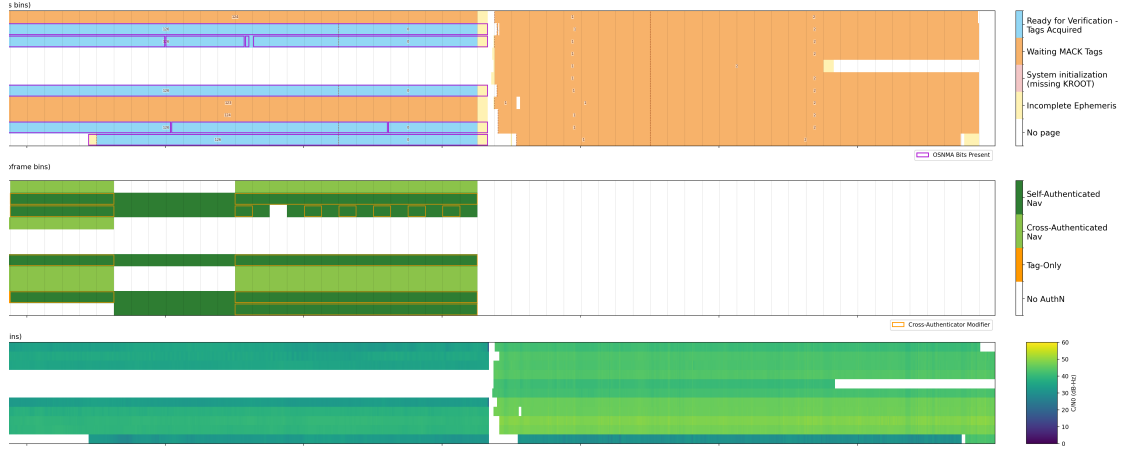


Figure 6.10: Full diagnostic output for the 17:30–18:00 recording: satellite extraction state, navigation-data authentication timeline and C/N_0 , 2s bins.

and hold a high C/N_0 throughout the second half.

6.5.4 Comparison With the Other Recordings

Table 6.3 places this recording alongside every other one used to characterise OSNMA behaviour in this thesis.

Table 6.3: Recordings used to characterise OSNMA behaviour across the thesis, by attack type and how ground truth was established.

Recording	Attack	Ground truth
Nominal (Chapter 5)	None	By construction (clear sky)
Constrained (Chapter 5)	None (environmental only)	By construction (obstructed sky)
ZED-X20P constrained (Section 6.3)	None (environmental only)	By construction
SDR spoofing (Section 6.4)	Spoofing, cable-fed	By construction (signal generator)
Jammertest live (this section)	Meaconing and spoofing, over-the-air	Campaign transmission plan

The progression is: no attack, a synthetic attack delivered without ever leaving the cable, and finally a declared attack received over the air. Only the last of the three recordings shows the sustained, total loss of authentication described in Section 6.5.1, which the nominal and constrained recordings never produce even under signal degradation. It shows that the two layers remain complementary under a real attack and not only under the environmental degradations of Chapter 5: the authentication state carries the event in full while the physical observables carry none of it. The models of the second part of this thesis operate on those physical observables alone, which is what makes their performance on this class of event a meaningful thing to measure.

Chapter 7

GNSS Interference Detection and Classification

7.1 Introduction

OSNMA authenticates navigation data. It does not observe the signal, and, as Chapter 5 showed, several degradations that destroy a position solution leave the authentication state untouched, while at least one genuinely suspicious event leaves the physical layer untouched. The second part of this thesis therefore addresses the complementary question: whether the receiver observables alone carry enough structure to tell a nominal interval from an interfered one. Two questions are asked of the observables. The first is whether the E1 signal at a given moment departed from nominal behaviour, which is a detection problem. The second is which signal was transmitted, which is a classification problem with three classes, nominal against jamming against spoofing, where spoofing covers every non-jamming transmission of the campaign, that is, meaconing, time offset, pseudorange error, position jump and navigation-data manipulation[49]. The two problems differ in what they need and in what they can deliver (Table 7.1).

Table 7.1: The two models side by side.

	Detector	Classifier
Question	Is this nominal?	Which of the three?
Labels in training	excluded	required
Objective	reconstruction	cross-entropy
Generalisation	unseen attack types	trained classes only

This chapter first reviews where machine learning has been applied to GNSS

threats and which constraints such a system has to satisfy. It then states the expected effect of interference on the receiver observables, and closes with the principle of the two families of models used in this work.

7.2 Machine Learning for GNSS Threats

Existing work does not address GNSS threats at a single point of the receiver chain. It is more useful to organise it by the layer of the chain at which the detection input is taken, because the layer determines what can be seen.

Physical layer. The input is the raw signal and the shape of the correlation peak. Convolutional networks are the usual choice, since the task is essentially the recognition of a deformation in a two-dimensional acquisition surface caused by a counterfeit signal overlapping the authentic one[50, 11]. The distinguishing property of this layer is that detection happens before any navigation data are extracted.

Observables layer. The input is the set of quantities the receiver produces per epoch and per satellite: pseudoranges, Doppler shifts and the behaviour of the local clock. Here the sequence carries the information, so recurrent models, and LSTMs in particular, are the standard approach [13, 38, 14]. The model learns the natural evolution of the receiver clock and of the measurement geometry, and flags movements or time jumps that are physically implausible. The detector and the classifier developed in this thesis operate at this layer.

Logical layer. The input is the OSNMA protocol itself: authentication timings, transitions of the receiver state machine, and the rate at which expected material fails to arrive. Hidden Markov Models and tree ensembles are candidates for describing the nominal flow of a finite state machine and predicting its next state. The target is protocol-level manipulation, such as an attacker intercepting and delaying otherwise legitimate data to drive the state machine into a state of its choosing. The literature reviewed offers no direct precedent for applying this class of models to the OSNMA state machine; the diagnostic outputs of Chapter 5 are the material such a model would consume.

Cross-layer. The input is everything above, combined with sensors that do not depend on the radio-frequency channel, such as an inertial unit or odometry[37]. The goal is consistency checking instead of detection in a single domain: a successful OSNMA verification coupled with a Doppler pattern incompatible with the motion of the platform is an inconsistency that neither layer would report on its own.

7.3 Design Constraints

Several constraints emerge from the literature and shape what a deployable system may look like. They are stated here because they justify several choices made in the following chapters.

Unsupervised learning for unseen attacks. A model trained on specific attack patterns recognises those patterns. Learning the safe baseline of the environment instead, and flagging departures from it, is what allows an attack never seen during training to be detected [50, 15]. This is the reason the detector used in this thesis never receives an attack label.

Environmental specificity and federated learning. Every environment produces different nominal values for the same feature, so what counts as normal is relative to the location. Models must therefore be tailored to comparable conditions, and collaborative training across receivers is meaningful only among local models trained in similar environments[51, 52].

Human in the loop and resilience. In critical operations, missed detections are as damaging as false alarms, so human verification remains necessary; but suspending the service to investigate is itself a self-inflicted denial of service. Explainable outputs, readable quickly by an operator, are one mitigation[53], and a lower-accuracy fallback source should be available for the intervals in which GNSS is not trusted [37].

Computational placement. Running the models on the receiver chip constrains the number of parameters and layers, and requires keeping enough capacity to catch a careful attacker. Running them offline at network level relaxes the computational budget at the cost of immediacy [16].

Continuous adaptation. Environments change, especially in urban areas, so the baseline itself has to be updated over time to avoid an accumulation of false alarms.

7.4 Interference Signatures

Each kind of interference acts on the receiver through a different physical mechanism, and therefore leaves a different joint signature on four quantities: the front-end gain set by the AGC, the noise level measured by the GNSS core, the average C/N_0 across satellites, and the dispersion of C/N_0 across satellites. Table 7.2 states the *expected* joint signature of each scenario, including two benign degradations that a detector must not confuse with an attack.

Table 7.2: Expected joint signatures.

Scenario	agc	noise	cn0	cn0_var	Mechanism
Wideband jammer	↓↓	↑↑	↓↓	~	Broadband power raises the noise floor; despread SNR falls for all satellites alike, so the spread is unchanged.
Narrowband CW	↓	~	~	~	The gain responds to total in-band power, but correlation with the local replica spreads an unrelated carrier before it reaches the C/N_0 estimate.
Power-matched spoofer	~	~	↑	↓↓	Counterfeit power adds fractions of a dB before despreading; afterwards it correlates strongly and one transmit geometry collapses the spread.
High-power spoofer	↓	~	↑↑	↓↓	Same mechanism with enough power to move the gain, hence far easier to catch.
Antenna degradation	↑	~	↓	~	Worse noise figure, less signal at the front end: the only case where the gain rises. Benign, slow.
Sky obstruction	~	~	↓	↑	Low-elevation satellites attenuated, high ones not: mean falls, spread widens, opposite to spoofing.

7.5 Anomaly Detection

The detector is an LSTM autoencoder: an LSTM encoder, an LSTM decoder, MSE loss, Adam optimiser. The anomaly score of a window is its reconstruction error, computed on the physical features only and not on the validity masks.

The setting is best described as self-supervised in its objective and one-class in

its training *modus operandi*. No attack label enters the fit, and the reconstruction target is derived from the input itself; the model is nevertheless trained on nominal rows only, and the alarm threshold is chosen on a labelled validation block, never on test. Three elements therefore define it: input without labels, reconstruction error as the score, and a threshold calibrated on labelled validation data.

To establish whether the temporal context is what produces the result, the same method can be applied without the temporal model. A One-Class SVM is trained in the same way: nominal rows only, no attack label in the fit, one row per epoch in and one score per epoch out. Its score is a distance from a learned boundary instead of a reconstruction error, and comparing it with the autoencoder isolates the contribution of the temporal information.

7.6 Attack Classification

The classifier is a unidirectional LSTM trained with a class-weighted cross-entropy in a many-to-one arrangement: a sequence of timesteps in, the last hidden state passed to the classifier head. The question it answers is therefore "given the preceding seconds, what is the state now?". Since the attack classes are rare with respect to nominal, class imbalance has to be handled explicitly, both in the loss during training and in the decision rule at inference time.

As for the detector, a memoryless counterpart isolates what the sequence contributes: a supervised Random Forest reading one epoch at a time, on the same three classes and the same partitions.

Chapter 8

Experimental Setup and Scenarios

8.1 Objectives

The experiments of this part of the thesis address three questions, all on the receiver observables alone and all on data recorded on days that the models have not seen during training:

1. whether a model trained without attack labels detects a departure from nominal reception;
2. whether a supervised model identifies the transmitted attack, distinguishing nominal, jamming and spoofing;
3. whether the temporal context read by a sequence model improves on a memoryless model trained under identical conditions.

This chapter describes the data, the models and the evaluation protocol used to answer them; the results are reported in Chapter 9.

8.2 Jammertest 2025

The labelled data come from the Jammertest 2025 campaign at Andøya [49]. The recording used here was made with a static Septentrio receiver at a single site, on Galileo E1 only. Roughly three days of nominal recording made at the same position, before the campaign schedule started, precede the declared experiments. The recording site is Test Area 1, and is unaffected by transmissions from the other test areas.

8.3 Dataset

8.3.1 Extraction

Two extraction scripts were written, one per receiver family, both producing a single timestamp-ordered CSV that serves as the training-dataset source.

`extract_sbf_params.py` and `extract_ubx_params.py` perform the same task on Septentrio SBF and U-blox UBXbinary logs. Building the dataset from the Andoya campaign, whose raw files were recorded using Septentrio receiver, required handling Type-2 **MeasEpoch** sub-blocks, sign extension of the offset fields, the C/N_0 offset correction, and taking elevation and azimuth from **ChannelStatus** instead of **SatVisibility** like in UBX.

The resulting dataset has one row per epoch, which is the level at which the cross-satellite statistics are defined. It covers about five days of recording, with 15.9% of the rows falling inside a declared transmission.

8.3.2 Labels

The ground truth is built from the official transmission plan: a row is labelled as attacked when its timestamp falls inside a declared transmission period. Label alignment was additionally verified through cross-correlation on the AGC signal. This binary labelling is the one used by the detector.

For the classifier, the aggregated recording is re-labelled for the three-class task: one row per second. The label set was refined relative to the binary ground truth: transmissions whose effect on E1 is ambiguous were removed, and declared events with no measurable effect on E1 were labelled back as nominal.

Table 8.1: Class composition after re-labelling.

Label	Class	Share
0	nominal	90.1%
1	jamming	3.9%
2	spoofing	3.4%
12	mixed jamming + spoofing	2.6%

Mixed events arise from the more complex tests of the transmission plan, where a window can contain both classes concurrently. They keep their own label for evaluation and are excluded from training, since there is no correct target for them.

Before training, the labels were checked against the data: does each labelled interval contain the effect it claims, or is the label wider than the attack? A nominal reference state was computed from every epoch that is labelled nominal, tracks at least one satellite, and lies more than 3 600 s away from any labelled transmission:

302 251 epochs. For each feature the reference is the median and a robust sigma, taken as $\text{MAD} \times 1.4826$ (Table 8.2).

Table 8.2: Nominal reference state.

Feature	Median	Robust σ
agcCnt_L1	55.00	5.94
noisePerMS_L1	100.00	2.24
cn0_mean	36.60	1.82
cn0_var	13.70	14.02
delta_dop_var	0.06	0.04

A row is counted as showing an effect if the receiver tracks nothing, or if any of those five features lies beyond 3σ from the reference; the coverage of an event is the fraction of its seconds showing an effect. The lowest coverage observed is 77.3%, and from the third lowest event onwards every remaining event exceeds 99%. No interval needed narrowing and no event had to be discarded for label noise.

Total denial, that is `n_tracked` equal to zero, is strongly class-dependent: it occurs in 5.2% of nominal epochs, 83.5% of jamming, 27.5% of spoofing and 29.0% of mixed.

8.3.3 Features and Preprocessing

Besides the AGC, the extraction pipeline computes quantities that describe the cross-satellite structure of the constellation: the mean and the dispersion of C/N_0 across satellites, the spread across satellites of the epoch-to-epoch Doppler change, the measured noise level, the number of tracked satellites and the receiver clock drift. The features kept are those that respond to interference through a physical mechanism and that are independent of the observation geometry, each responding to a distinct mechanism of Section 7.4. Geometry is excluded because, for a static antenna, the triple of satellite identifier, elevation and azimuth identifies one instant of the day, so a model reading it could learn when a transmission occurred rather than what was received. The features are summarised in Table 8.3.

The classifier uses a seventh feature, the receiver clock drift `clkD`, expressed in nanoseconds of drift per second, added as a candidate channel for time and pseudorange manipulation, which the power and front-end features do not measure directly. Its contribution has not been demonstrated yet and a leave-one-feature-out ablation is the natural test. Four features can be missing, each contributing a validity mask, so the per-epoch input of the classifier is 11 channels.

The preprocessing steps, common to the detector and the classifier unless stated otherwise, are summarised in Table 8.4.

Table 8.3: Retained features and their role.

Feature	Scope		What it captures
agcCnt	per block	RF	Front-end gain, set before the ADC. Any added in-band power, deliberate or not, forces the gain down.
noisePerMS	per block	RF	Noise level measured by the GNSS core. Complementary to the AGC; read as a pair the two separate jamming from spoofing.
cn0_mean	global		Average received power. Falls under broadband jamming, rises under high-power spoofing, and drifts slowly with antenna or cable degradation.
cn0_var	global		Dispersion of C/N_0 across satellites. An authentic constellation varies with elevation; a single-antenna spoofer makes the levels uniform and the dispersion collapses.
delta_dop_var	global		Spread across satellites of the epoch-to-epoch Doppler change, which should be smooth for a fixed antenna. Authentic Doppler is mutually independent across satellites; signals injected from one transmitter share a single geometry and become correlated.
n_tracked	global		Number of tracked satellites; drops under jamming.

8.4 Models

8.4.1 Detector

The configuration of the autoencoder is reported in Table 8.5. It reads the six aggregate features of Table 8.3, excluding the clock drift, together with the validity masks of the three that can be missing, and reconstructs all nine columns.

8.4.2 Classifier

Three terms are used for the classifier. An *epoch* is one second, one row of the dataset. A *window* is 60 consecutive epochs, a hyperparameter, and is what the model reads and predicts. An *event* is a contiguous interval carrying one label, and is what gets assigned to a split. Events are the unit of assignment, windows the unit of training and prediction: an event is placed whole into train, validation or

Table 8.4: Preprocessing steps and their motivation.

Step	What	Why
Session cleaning	correlation-based filtering of redundant features, session detection, trimming of receiver warm-up intervals	Removes redundant inputs and the transient at receiver start.
Absolute time excluded	week, TOW and absolute seconds used only for ordering	Jamming and spoofing sit on different campaign days; a clock input would let the model complete the task by reading the calendar.
Tail compression	$\log_{10}(x + 10^{-3})$ on $\Delta\text{Doppler}$ variance	Normalises the skew where the mass sits below 1; a plain $\log(1 + x)$ would be close to the identity there.
Validity masks	one per feature that can be missing	Tells the network which values were NaN-replaced. Computed from the data only, never from the label.
Missing values	training-split median	—
Standardisation	fitted on the training split only	—
Segmentation on gaps	any hole in the second-by-second index ends a segment	A window bridging a gap would teach the network that a discontinuity is normal.

Table 8.5: Configuration of the autoencoder.

Window	60 s
Stride	5 s
Latent size	32
Input	6 features + 3 masks
Reconstructed	9 columns

test, and the windows built from its epochs inherit that placement.

Windows are 60 s long, with a training stride of 5 s as for the detector and an evaluation stride of 1 s, so that at test time the model produces one prediction per second, each based on the preceding sixty. The label of a window is the label

of its last timestep. The binary detector could use an "any attack epoch in the window" rule, because its question has two answers. With three classes that rule is undefined: a window spanning the end of a jamming event and the start of a spoofing one contains both. The last epoch is defined in every case and attaches the prediction to the present rather than to an average over the window.

A window is kept only if every timestep it covers belongs to the same split, which is what makes the partition leakage-free, while still allowing a window to cross a nominal to attack boundary when both events are on the same side. Those crossing windows are kept because a window is scored on its last epoch, requiring windows to lie entirely inside one event would make an event starting at t_0 unscorable before $t_0 + 59$.

This rule has a limitation. A window ending exactly at t_0 carries the attack label while its content is 59 seconds of nominal and one second of attack. The target is correct, yet the input barely supports it, so in training such onset windows behave like label noise: the only way to fit them is to react to a single anomalous epoch, which is also what produces false alarms on nominal transients. Per-epoch supervision, returning all 60 hidden states and computing the loss against 60 targets, is the natural alternative and is left as future work.

The network has one LSTM layer with 32 hidden units, trained with Adam, with a 60×11 input window. Class imbalance is handled at two points. During training, inverse-frequency weights in the loss make a rare class count more. At decision time, each of the three probabilities is multiplied by a per-class coefficient before the largest is taken; nominal is fixed at 1, since only the ratios matter, which leaves two coefficients to search. Every combination on a grid from 0.25 to 16 is evaluated on the validation block and the pair maximising macro-F1 is kept.

Total-denial windows are kept in training. A window in which all 60 epochs have zero tracked satellites still contains 60 epochs of AGC and noise floor. A direct ablation, with identical splits, hyperparameters and threshold selection and only the training set changed, gives a jamming F1 over all test windows of 0.136–0.453 when those windows are removed against 0.599–0.913 when they are kept. Over non-denial windows the two are comparable in three of the four draws where jamming F1 is defined (0.906 / 0.734 / 0.740 removed against 0.858 / 0.657 / 0.703 kept) and worse in the fourth (0.345 against 0.621). Keeping them therefore does not improve performance where the receiver is tracking, but makes denial windows classifiable, and those are the larger part of the jamming class.

8.4.3 Baselines

Each sequence model is paired with a memoryless baseline trained under the same protocol, which serves as the reference against which its results are measured and isolates the contribution of the temporal context: a One-Class SVM for the detector

and a Random Forest for the classifier (Tables 8.6 and 8.7). The One-Class SVM cannot serve for the classifier, since it answers a different question on a different partition.

Table 8.6: One-Class SVM configuration.

Kernel	RBF
ν	0.05
γ	scale (inputs already standardised)
Input	9 values per epoch, no window
Score	<code>–decision_function</code>
Threshold	F1-optimal on the validation block

Table 8.7: Random Forest baseline configuration.

Trees	300
<code>max_depth</code>	unlimited
<code>min_samples_leaf</code>	5
<code>class_weight</code>	balanced
Input	11 values per epoch, no window
Decision	validation-fitted class multipliers

For the One-Class SVM, the sign of the score is flipped so that larger means more anomalous, as for the reconstruction error, leaving the alarm rule and every metric unchanged. The parameter ν is the fraction of training points allowed outside the boundary, and since the training slice is nominal by construction it only shapes that boundary; the operating point is chosen afterwards, on validation. For the Random Forest, a leaf size of 5 is imposed because at 1 Hz neighbouring epochs are near duplicates and leaves of size one would most probably memorise them; depth is left unlimited because train and test already hold different events by construction.

Both baselines read the preprocessing directories that the LSTM runs already produced, so the partitions are the same ones, fold by fold and seed by seed. Identical across each pair are the feature set and validity masks, the filling of missing values with the training-split median, the standardisation fitted on the training split only, the segmentation rule on gaps, the partition itself, the rule used to set the operating point on the validation split, and the reporting code. For the classifier pair, in addition, `class_weight = balanced` computes the same inverse frequencies as the LSTM loss weights, the decision multipliers are searched on the same 0.25–16 grid, mixed events are scored and never trained on, and denial rows stay in training.

8.5 Evaluation

8.5.1 Detector Protocol

The recording is divided into four blocks, detected automatically by cutting at the midpoint of every gap longer than six hours between consecutive declared transmissions. Any value between 4.87 and 11.00 hours produces the same four blocks, which correspond roughly to campaign days.

Each block takes a turn as the test set; another block acts as validation; the nominal rows of the remaining blocks are the training data. Attack rows inside the training blocks are discarded, because they cannot train a model that learns nominal behaviour only, and scoring them would mean evaluating attacks surrounded by data the model has already seen. Each of them is evaluated in another fold, when its own block takes its turn as test. This gives twelve ordered (test, validation) pairs.

A single chronological split (for example 70/15/15) was rejected for three reasons. The attacks inside the training window cannot be recovered without an arbitrary reinsertion rule that reintroduces leakage in any case. The test set would be the last day only, so the result would hold for one day out of four. And a single split returns a single number with no measure of its variability.

8.5.2 Classifier Protocol

The detector’s leave-one-block-out protocol cannot be reused. It trains on nominal only and sends every attack to test, which is not a valid arrangement for supervised classification; and its blocks are campaign days, while the campaign placed jamming on Monday and Tuesday and spoofing from Wednesday to Friday, so holding out a day removes a whole class from training. An event-level split is used instead.

An event is defined as a maximal run of constant label and usability flag inside one gap-free segment. Nominal runs are then cut into 1 800 s blocks, so that there are enough nominal events to distribute; two additional cuts were placed by hand inside the long Tuesday jamming block, at its two emitter-change boundaries[49]. The result is 12 usable jamming events, 17 usable spoofing events, and nominal divided into blocks.

Neither a row-level nor a window-level random assignment is suitable. With a stride smaller than the window, two consecutive windows share most of their epochs, so a random window assignment puts near-duplicates on both sides; a random row assignment has the same effect indirectly, since windows are built from rows drawn from both sides. Assigning whole events keeps all the rows describing one transmission on the same side of the partition.

Assignment is performed per class independently: the events of a class are

shuffled with a seeded generator and assigned by event count (60% train, 20% validation, 20% test), not by duration, since jamming events range from a few tens of seconds to 10 792 s and equal counts would not mean equal support. The draw is then checked against a support constraint.

Because 83.5% of jamming epochs are total denial, an event count alone is not a sufficient constraint. The rule adopted requires each class to place, in each of train, validation and test, at least $\max(120 \text{ s}, 12\% \text{ of that class's own tracking budget})$ of non-denial windows. Five seeded draws satisfying the constraint were produced. Each draw differs only in the seed used for the event-to-split assignment; model, features and hyperparameters are identical across the five.

8.5.3 Metrics

The share of attacked rows differs by up to a factor of ten across test partitions, so raw PR-AUC is not comparable between them. The headline metrics are therefore ROC-AUC and PR-AUC divided by its own baseline, the share of positives, which expresses the factor gained over a random scorer. F1 depends on the operating point, and is always reported as a range over the calibrations and next to the share of attacked rows in its own partition. Event recall, the fraction of labelled events in which at least one alarm is raised, is reported for the detector and its baseline.

Total-denial units are treated explicitly. For the detector, total-denial epochs are excluded from evaluation. For the classifier, results are reported twice: over all windows, and over windows that are not entirely inside a denial block. The second figure is the one comparable with the detector's, and the gap between the two states explicitly how much of the score rests on windows with no signal content.

Not everything is comparable between a sequence model and its memoryless baseline. Event recall is, since an event is caught or it is not whatever the unit. PR-AUC over its own baseline is, as a ratio; once total-denial units are removed from both sides, the share of attacked units is the same and the ratio is on a common scale. Per-class F1 as an absolute number is not, being per epoch on one side and per window on the other, each model at its own fitted operating point. Macro-F1 and anything computed on validation are not usable for the comparison, since validation is where both models fitted their operating point.

Chapter 9

Results and Discussion

This chapter reports the results of the experiments described in Chapter 8: first the detector, then the classifier, then the comparison of both with their memoryless baselines. The findings are summarised in Chapter 10.

9.1 Anomaly Detection

Table 9.1 reports the test-block figures, averaged over the three validation blocks of each fold, with total-denial epochs excluded. The attack-class F1 is given as the range over those three calibrations rather than as a single value, for the reason developed in Section 9.1.1.

Table 9.1: LSTM detector, leave-one-block-out, denial epochs excluded.

Test block	ROC-AUC	PR-AUC	PR-AUC baseline	Attack F1 range
1	0.901	0.823	0.13	0.340 – 0.754
2	0.982	0.929	0.30	0.927 – 0.948
3	0.984	0.875	0.10	0.383 – 0.769
0	0.330	0.021	0.03	0.021 – 0.025

On the scale of PR-AUC over its baseline, the three campaign blocks give 6.3, 3.1 and 8.8 times the random scorer. No fold is the best one: block 2 leads on raw F1 only because it contains the largest share of attacked rows.

9.1.1 Threshold Transfer

Keeping the test data and the model fixed and changing only the validation block used for calibration produces Table 9.2.

Table 9.2: Spread induced by the choice of validation block.

Test block	ROC-AUC spread	F1 spread
0	0.068	0.005
1	0.018	0.414
2	0.004	0.021
3	0.022	0.387

Threshold-free metrics barely move, while F1 moves by up to 0.41 on identical test data. On test block 1 the score is 0.340 when calibrating on block 0 and 0.754 when calibrating on block 2, with the same model and the same reconstruction ability.

A fixed percentile threshold is not a deployable design, since the reported F1 depends on which block calibrated it as much as on the model. The score itself is stable and transfers across days, which suggests adaptive normalisation, for example a rolling median and MAD computable online without labels, intrad of a fixed operating point.

9.1.2 Block 0

On block 0 the detector inverts. A ROC-AUC of 0.330 is below 0.5, which means the ranking is reversed: the pre-campaign nominal data receive a higher reconstruction error than the attacks themselves. Block 0 is primarily made of the recording made before the campaign schedule started, and its nominal behaviour is not drawn from the same distribution as the campaign days. This explains both the inverted score and the poor calibration it provides to the other folds: used as the validation block, it returns a threshold that is too high and does not transfer.

9.2 Attack Classification

The test split contains only classes 0, 1 and 2, since mixed events never appear there. Table 9.3 reports all test windows, Table 9.4 excludes the all-denial windows.

Table 9.3: Classifier, all test windows.

Draw	Jam F1	Jam PR-AUC	base	Spoof F1	Spoof PR-AUC	base	Macro-F1
0	0.913	0.984	0.056	0.626	0.614	0.025	0.842
1	0.894	0.771	0.092	0.565	0.336	0.033	0.812
2	0.599	0.755	0.028	0.793	0.833	0.037	0.793
3	0.667	0.688	0.026	0.861	0.957	0.034	0.840
4	0.831	0.949	0.058	0.771	0.937	0.051	0.865

Table 9.4: Classifier, excluding all-denial windows.

Draw	Jam F1	Jam PR-AUC	base	Spoof F1	Spoof PR-AUC	base	Macro-F1
0	0.858	0.862	0.008	0.680	0.634	0.026	0.842
1	—	0.049	0.007	0.558	0.395	0.036	0.769
2	0.657	0.735	0.022	0.870	0.852	0.039	0.838
3	0.703	0.730	0.021	0.918	0.967	0.034	0.871
4	0.621	0.665	0.011	0.871	0.957	0.045	0.829

Excluding draw 1, jamming F1 ranges from 0.62 to 0.86 and PR-AUC over baseline from $33\times$ to $108\times$. Spoofing is far more regular: over the same four draws its PR-AUC over baseline is $24\times$, $22\times$, $28\times$ and $21\times$, nearly constant. The instability is specific to jamming.

The mixed events, 10 896 windows never used in training, were also scored. There is no correct answer for them, so the predicted distribution is the result instead of an error rate: across draws it is 0.3–13.0% nominal, 9.4–23.6% jamming and 76.0–81.4% spoofing. Both attack classes appear, with spoofing dominant, which is consistent with their composition, generally spoofing preceded by an initial jamming phase, under a last-epoch labelling rule.

9.2.1 Jamming Failure Case

In draw 1 the jamming F1 over non-denial windows is undefined: all 542 non-denial jamming windows of that test split were classified as nominal, so recall is zero and precision undefined. In the same draw it reaches a jamming F1 of 0.766 on the validation split (804 windows) and 0.894 over all test windows (7684 windows). Those 542 windows are a single scenario: a continuous wave at 47 dBm transmitted simultaneously on five bands. Draw 0’s test split contains a comparable amount of non-denial jamming, 606 windows from roughly one scenario, and reaches 0.858. The two splits hold a comparable quantity of jamming but different scenarios.

Table 9.5: Measured effect on E1 of two 47 dBm CW scenarios.

	Nominal reference	Single-band CW	Five-band CW
agcCnt_L1	55.0	48–50	52–53
cn0_mean	36.6	30–32	26–27
Tracked satellites	—	10 $\rightarrow$ 8	11 $\rightarrow$ 9

Both transmit the same total power, but one concentrates it on L1 while the other divides it across five bands. On E1 the five-band case disturbs the front end less, lowers C/N_0 more, and the receiver keeps lock throughout. The pattern reproduces across draws: where that scenario is the only jamming present in an

evaluation split, the score is 0.000 (draw 1, test) and 0.148 (draw 4, validation); where it is mixed with other scenarios, or used for training, the score is normal (0.657, 0.858, 0.703).

Jamming performance is therefore not a single number: it depends on which scenario the model is evaluated against, and specifically the scenario it misses is the one that degrades signal quality without disturbing the front end or costing lock. This effect is visible only because the partition was drawn five times.

9.3 Temporal vs Memoryless Models

9.3.1 Detection

Table 9.6: One-Class SVM, leave-one-block-out, denial epochs excluded.

Test block	ROC-AUC	PR-AUC	Baseline	F1 range	Event recall
0	0.223	0.017	0.027	0.007 – 0.032	12–13 / 14
1	0.891	0.814	0.127	0.598 – 0.745	22–24 / 25
2	0.986	0.942	0.298	0.823 – 0.921	23–24 / 25
3	0.989	0.897	0.102	0.025 – 0.843	12–18 / 18

Table 9.7: Detector against One-Class SVM, denial epochs excluded.

Test block	ROC-AUC		PR-AUC over baseline	
	LSTM	OC-SVM	LSTM	OC-SVM
1	0.901	0.891	6.3×	6.4×
2	0.982	0.986	3.1×	3.2×
3	0.984	0.989	8.8×	8.7×
0	0.330	0.223	0.7×	0.6×

On the three campaign blocks the two models perform equivalently: ROC-AUC differs from LSTM to OC-SVM by at most 0.01 and the PR-AUC factor by at most 0.06, far inside the spread that changing the validation block produces for either of them.

On block 0 both invert, the baseline slightly further. Two architectures trained on the same nominal rows fail in the same way on the same block, which indicates that the cause is the data and not the sequence model.

A distance from a learned boundary has no stable scale across different folds, which reinforces the argument for a threshold-free metric.

9.3.2 Classification

Table 9.8: Random Forest, five draws, test excluding all-denial epochs. PR-AUC is given with the class share of that draw in brackets.

Draw	Jam F1	Jam PR-AUC	Spoof F1	Spoof PR-AUC	Macro-F1
0	0.743	0.892 (0.008)	0.926	0.806 (0.020)	0.888
1	0.061	0.539 (0.008)	0.721	0.417 (0.022)	0.590
2	0.694	0.872 (0.020)	0.946	0.973 (0.040)	0.878
3	0.737	0.795 (0.018)	0.926	0.881 (0.030)	0.886
4	0.306	0.519 (0.008)	0.888	0.970 (0.044)	0.726

Table 9.9: Classifier against Random Forest, means over the five draws, test excluding all-denial units.

	Random Forest	LSTM	Better
Spoofing F1	0.881	0.779	baseline
Spoofing PR-AUC over baseline	27×	21×	baseline
Jamming F1	0.508	0.568	neither, both poor
Jamming PR-AUC over baseline	66×	49×	baseline

Spoofing does not require a sequence model: a memoryless forest reading one epoch outperforms the LSTM on both measures, so whatever separates spoofing on this receiver is present in a single epoch’s features. Neither model handles jamming well: both swing by roughly 0.28 across draws, the LSTM’s minimal advantage being stability, with a macro-F1 standard deviation of 0.033 against 0.119. Both fail on draw 1, the one where the five-band scenario stands alone in test.

Chapter 10

Conclusions and Future Work

10.1 Summary

This thesis studied two complementary ways of monitoring Galileo reception on low-cost receivers. The first is protocol-level: the OSNMA workflow was implemented as an offline and live pipeline built on OSNMAlib and on a Galileo-only SPP solver adapted from CSSRlib, with diagnostic outputs that expose every intermediate authentication state on the same time axis as the receiver observables. The second is observation-level: an LSTM autoencoder and a three-class LSTM classifier were trained on the Jammertest 2025 recording, and each was compared with a memoryless model trained under the same protocol.

10.2 Main Findings

1. **The intermediate states of OSNMA explain events that look like errors.** Cross-authentication without local tags, waiting for a delayed key and the buffering of navigation data under degraded reception are nominal protocol behaviour. Reading the protocol state per satellite and per sub-frame, rather than as a pass-or-fail response, avoids interpreting them as failures, and provides an explanation that the physical observables cannot supply on their own.
2. **Enforcing authentication in the position solution has no cost in nominal conditions.** With a clear sky view the all-Galileo and authenticated-only solutions coincide. In the constrained scenario the two diverge, and the

discontinuity in the authenticated-only solution is directly traceable to the moment a satellite completes its authentication.

3. **OSNMA and the physical observables react to different problems.** Poor satellite geometry produced the largest positioning error observed while all authentication states remained nominal, because the decoded navigation data were correct. During a declared meaconing and spoofing transmission, instead, no satellite authenticated after the attack started, while the observables gave useful but ambiguous indications: C/N_0 increasing rather than decreasing, and satellites absent in nominal conditions appearing with strong signals. The observables flag the anomaly; only the authentication state confirms it.
4. **Given this specific Andoya dataset, the receiver observables alone are sufficient to detect an anomaly on days never seen in training.** On the three campaign blocks the detector reaches ROC-AUC between 0.90 and 0.98 and PR-AUC three to nine times its baseline (Section 9.1).
5. **Temporal context does not improve the result.** A One-Class SVM reading one epoch matches the autoencoder reading sixty within 0.01 ROC-AUC, and a Random Forest outperforms the LSTM classifier on spoofing (Section 9.3). The limiting factor is the available dataset and the feature set, if more sophisticated attacks are conducted and recorded, not the architecture.
6. **The score transfers across days; the alarm threshold does not.** On the same test data and with the same model, calibrating the threshold on different validation blocks moves F1 from 0.34 to 0.75 (Section 9.1.1). On the pre-campaign data both detector architectures invert in the same way, which points to a change in the nominal distribution rather than to a model failure (Section 9.1.2).

10.3 Limitations

- **Jamming coverage.** The jamming class contains 12 usable events, only 4 of which carry more than 120 tracking seconds, and groups signals of very different waveform, bandwidth and power. Five random draws do not guarantee that every scenario is tested.
- **Adjacent events.** One long jamming event was cut by hand at its two emitter-change boundaries. The resulting events share acquisition conditions, so figures computed over all-denial windows carry an optimistic bias.
- **Single campaign, site and receiver.** All labelled data come from one static Septentrio receiver at one site, and the ground truth is derived from the

declared transmission plan rather than from an independent measurement of the emitted signals.

- **Fixed operating point.** The thresholds used in this work are calibrated offline on a validation block, and do not transfer across days.

10.4 Future Work

Feature set. Since the architecture is not the limiting factor, the next improvements should come from the input. An ablation of the validity masks would establish whether they contribute, and a leave-one-feature-out test is needed for the receiver clock drift, whose contribution has not been demonstrated yet.

OSNMA-derived features. The outputs of the OSNMA pipeline, namely extraction states, authentication states, tag-source relations and the gap between tracked, complete and authenticated satellites, form a feature block that a purely physical detector cannot have. They were deliberately kept out until the physical-layer result was established, so that their contribution can now be measured rather than assumed, in particular on the jamming class and on the ambiguous cases observed under meaconing.

Jamming evaluation. Leave-one-scenario-out is the protocol the jamming class requires, together with more jamming recordings covering a wider range of waveforms and power levels.

Other datasets. The models learn the nominal conditions of the environment in which they are trained, so their transfer to other receivers, sites and campaigns can only be established by testing on other data. The SDR scenario of Chapter 6 provides a first controlled case with known ground truth.

Appendix A

OSNMAlib output README

Chapters 5 and 6 read the diagnostic outputs of the pipeline to argue about specific recordings, and each of those readings rests on the exact meaning of a colour, a field or an axis. This appendix reproduces, verbatim, the reference document distributed with the implementation, which states for every artefact the pipeline writes which source file each quantity comes from, how a state is decided, and what a single plotted cell represents. It contains no findings of its own.

```
# OSNMA Timelines
```

```
The timelines in plot_results.py merge four info sources:
```

- ****Verbose text log: general_logs.log**** (fallback timing source)
 - This file is created by the logging setup in logger_factory.py.
 - TTFF/TTFAP are emitted by the navigation/auth logic in ``nav_data_manager.py`` at the **moment the event is detected**.
 - Concretely, when the receiver reaches "first fix" (TTFF) or "first authenticated fix" (TTFAP), ``nav_data_manager.py`` logs lines like:
 - ``First Fix at GST <WN> <TOW>``
 - ``First Authenticated Fix at GST <WN> <TOW>``
 - ``First GST <WN> <TOW>`` (the reference start time)
 - ``TTFF <N> seconds``
 - ``TTFAP <N> seconds``
- Note: internally TTFF/TTFAP are computed at **page end** (``gst_page_end = LAST_GST + 2s``), because a full I/NAV page spans 2 seconds; this is why the "First ... at GST ..." TOW is typically aligned to the page boundary.

- ``plot_results.py`` parses these lines with regex in ``_load_ttffaf_from_general_logs()`` and uses them only as a ****fallback**** when ``run_metadata.json`` is missing those values or contains ``null``.
- ****run_metadata.json**** (preferred timing source + run flags)
 - Written by the runners (e.g. `run_file.py`). Fields:
 - `ubx_clock_messages`: whether trusted clock/time messages were used
 - `ttff`, `ttfaf`: durations in seconds (if computed)
 - `first_tow`: TOW of the first processed page (start reference)
 - `faf_tow`: TOW of the "first authenticated fix" event (if known)
 - The timeline code uses this to place and label the vertical TTFAF marker as accurately as possible:
 - 1) ****Best case: `faf\_tow` -> exact x-position****
 - If ``run_metadata.json`` contains ``faf_tow`` (TOW of the *first authenticated fix event*), ``plot_results.py`` builds a continuous 2-second time axis (``time_keys``) and searches for the sample whose TOW equals ``faf_tow``.
 - If the exact TOW sample is not present (e.g., the plotted range does not include it, or the axis is sparse/trimmed), it falls back to the ****nearest**** available sample in TOW to still draw the marker rather than dropping it.
 - 2) ****Next: `first\_tow + ttfaf` (mod week) -> derived x-position****
 - If ``faf_tow`` is missing but both ``first_tow`` (TOW of the first processed page) and ``ttfaf`` (seconds) are available, the code derives the event TOW as:
``faf_tow = (first_tow + ttfaf) % 604800``
 - Then it applies the same "exact match else nearest sample" logic as above.
 - 3) ****Else: heuristic from `status\_log.json` (subframe-level approximation)****
 - If numeric metadata is not available (which should virtually never happen), the code estimates the TTFAF moment from ``status_log.json`` by finding the first subframe where at least 4 satellites have authenticated ADKD0/ADKD12 (a practical definition of "first authenticated fix").
 - The marker is then placed on the first 2-second bin whose ``(WN, subframe_tow)`` matches that subframe.

- This is necessarily coarser (30 s resolution) than ``faf_tow`` and should be considered an approximation.
- If none of the above can be resolved, no TTFAF marker is drawn.
- ****Per-subframe status: status_log.json**** (auth & data state, 30 s frequency)
 - Produced by `status_logger.py` once per 30-second subframe (GST_subframe), after the receiver finishes that subframe in `receiver.py`. The plot code extracts:
 - `verified_OSNMA_material.tags` --> which satellites had verified tags (and tag source mapping `prn_d <- prn_a`)
 - `authenticated_nav_data` --> which satellites have authenticated nav blocks (ADKD0/4/12)
 - `nav_data_received` --> whether each satellite has a complete clear-nav block yet (words received)
 - `OSNMA_status.tesla_chain_in_force` and `.public_key_in_force` --> whether the system is "ready" (KROOT/PKR in force)
 - A heuristic TTFAF subframe: "first subframe where at least 4 satellites have authenticated ADKD0/ADKD12".
- ****Per-page data: page_log.csv**** (timeline grid source)
 - See Section 3 for full details. In the context of the Extraction State Timeline:
 - ****Rows (y-axis)**** is the sorted unique ``svid`` values observed in ``page_log.csv``.
 - ****Columns (x-axis)**** is a continuous time axis in ****2-second bins**** from the minimum to the maximum ``(wn, tow)`` found in ``page_log.csv``.
 - A cell is ****white / NLOS**** when ***no row exists in ``page_log.csv`` for that ``(svid, wn, tow)`` bin. This is why a satellite that never appears in ``page_log.csv`` cannot appear in this timeline at all.**

1) Satellite Extraction State Timeline

This timeline represents the receiver synchronization and extraction states during the TESLA protocol authentication process.

Additionally, it overlays Received IOD (Issue of Data) labels. These labels are extracted directly from the raw navigation data blocks (`nav_data_received`) and are displayed as soon as the satellite begins providing partial or complete navigation data (`state >= Incomplete Ephemeris`), regardless of their authentication status. Binary IOD strings are converted to decimal for display.

- ****(White) No Page****: No valid I/NAV page was processed for this satellite during this 2-second interval. This could be due to a failed CRC check, a dummy page, an alert page, or because the receiver is tracking a signal other than E1-B. No ``page_log.csv`` row for that bin. This state is also assigned when there is no ``page_log.csv`` row for that ``(svid, wn, tow)`` bin.
- ****(Yellow) Incomplete Ephemeris****: During this subframe period, the receiver successfully initialized the system (KROOT and MERKLE TREE acquired) but only extracted partial or malformed ADKDO navigation data from this satellite. Therefore, it is still waiting to assemble a complete set of navigation words. This state is assigned when:
 - chain + public key are in force, BUT ``nav_data_received[svid]`` is not complete (when all words 1..5 marked ``True``).
- ****(Pink) System Initialization / Missing KROOT****: Complete ADKDO navigation data has already been fully extracted from this subframe for this satellite. However, at the end of this subframe period, the receiver has not yet fully assembled the TESLA Root Key (KROOT). It must wait to collect the remaining DSM blocks in the subsequent subframes. This state is assigned when:
 - ``OSNMA_status.tesla_chain_in_force`` is ``null`` OR ``OSNMA_status.public_key_in_force`` is ``null``.
- ****(Orange) Waiting for Tag****: During this subframe period, complete ephemerides (all ADKDO words marked ``True``) were successfully extracted from this satellite, but no MACK tags were transmitted by this satellite in this specific subframe. This state is assigned when:
 - chain + public key in force, AND nav block is complete, BUT the satellite is **not** in ``mack_tagged_svids`` for that subframe.Meaning: we have the nav data, but we haven't seen any tag objects coming from this satellite's MACK yet.
- ****(Light Blue) Ready for Verification / Tags Acquired****: During this subframe period, complete ephemerides were extracted, and this satellite also actively transmitted MACK tags (regardless of whether these tags will eventually be used or verified). This state is assigned when:
 - chain + public key in force, AND nav block is complete, AND ``mack_tagged_svids`` contains this ``svid`` for that subframe.
 - ``mack_tagged_svids`` is derived from ``OSNMA_material_received[svid].mack_data.tags`` being non-empty (tag objects were extracted).Meaning: The satellite has provided both complete navigation data and its corresponding cryptographic tags. The acquisition phase is complete, and the data is queued, ready to be verified as soon as the protocol's disclosure delay passes and the TESLA key is received.
- ****(Violet Outline Modifier (OSNMA Present)****: The page's OSNMA material is actively present. This state is assigned when at least one bit among the 40 OSNMA bits is different from 0 (``has_osnma == True``).

2) Nav-Data Authentication Timeline

This timeline categorizes the actual data content and authentication payload contained within a specific subframe.

Unlike the Extraction State Timeline, this plot is evaluated purely in 30-second subframe blocks. If an authentication state is achieved for a subframe, the entire 30-second block (mapped visually to 2-second contiguous cells) will reflect that state, representing the protocol's delayed disclosure mechanics.

The resulting color states are:

- **White) No AuthN**: The satellite has no authenticated navigation data during this subframe's 30-second period.
- **Orange) Tag-Only**: During this subframe's 30-second period at least a tag, sent by this satellite's previous subframe, was used to authenticate navigation data of other satellites. However, this satellite itself does not have any authenticated navigation data during this subframe's 30-second period.
- **Light Green) Cross-Authenticated Nav**: Navigation data for this satellite was successfully cross-authenticated within this subframe's 30-seconds period. This is achieved using a TESLA key extracted in this current subframe's 30-seconds period (by any available satellite, not necessarily by this satellite) that authenticated a tag broadcasted in one of the previous other satellite's subframes (not by this satellite).
 - **Orange Outline Modifier (Cross-Authenticator)**: If this state features a orange outline, it indicates that this satellite's navigation data was authenticated via other satellites' tags, plus at least a tag received from this satellite in the previous subframe was successfully used in this current subframe's 30-seconds period to authenticate navigation data of other satellites.
- **Dark Green) Self-Authenticated Nav**: Navigation data for this satellite was successfully authenticated during this subframe's 30-seconds period. This is achieved using a TESLA key extracted in this current subframe's 30-seconds period that authenticated a tag contained in this satellite's previous subframe. Svid is in tags_verified and svid is in nav_authenticated.
 - **Orange Outline Modifier (Cross-Authenticator)**: If this state features a orange outline, it indicates that this satellite's navigation data was authenticated via this satellite's tag, plus at least a tag received from this satellite in the previous subframe was successfully used in this current subframe's 30-seconds period to authenticate navigation data of other satellites.

3) CSV Outputs

These files are generated inside each `logs\_` folder and are meant to make the plots easier to interpret and reuse.

- **``page\_log.csv``**: It contains one row for each received 2-second page and includes the satellite ID, GST week, GST TOW, subframe TOW, and the OSNMA-related flags needed to build the acquisition and authentication timelines. It determines which satellites exist on the y-axis and which 2-second time bins (wn, tow) appear on the x-axis of the4 extraction state timeline.

A CSV row is emitted only if all these conditions hold:

- **``Logging is enabled``**: `Config.LOG_FILE`` must be ``True`` (otherwise ``PageLogger`` never opens the CSV).
- **``The page exists in the input stream``**: the receiver must actually iterate an RXM-SFRBX page for that SVID. (A satellite can be present in NAV-SAT / skyplot but still never produce usable RXM-SFRBX pages.)
- **``It passes `_filter_page()```** (see ``receiver.py``):
 - band must be ``E1B``,
 - it must not be an alert page or a dummy page,
 - CRC must be OK,
 - it must not be earlier than the configured start time.
- **``It passes `_time_sync()```**: GST time must not go backwards.

What *does not* prevent a row from being logged:

- A subframe being later discarded for OSNMA processing reasons (e.g., missing NMA header, ``CPKS==0``, ``NMAS==0``) happens at end-of-subframe handling, *after* the individual pages were already written to ``page_log.csv``. Conceptually ``page_log.csv`` answers "which 2-second pages did the pipeline accept and attempt to process?" while ``status_log.json`` answers "what OSNMA state/authentication results did we have at the end of each 30-second subframe?"
- **``navsat.csv``**: This file contains the NAV-SAT extraction results. It lists the visible satellites together with the skyplot information, such as azimuth and elevation, so it is used to draw the skyplot and inspect the satellite geometry during the run.
- **``tag\_sources\_by\_subframe.csv``**: For each subframe and each authenticated destination satellite ``prn_d``, it shows which source satellites ``prn_a`` provided verified tags that were used to authenticate that satellite's navigation data. The file also includes the number of sources, if the destination satellite authenticated itself, and if the navigation data for that satellite is authenticated in that subframe.
 - **``wn` / `tow\_sf``**: GST week and subframe TOW that identify the subframe.
 - **``prn\_d``**: Destination satellite, that is, the satellite whose navigation data is being authenticated.
 - **``auth\_sources``**: The list of source satellites whose verified tags contributed to the authentication of ``prn_d``.
 - **``source\_count``**: How many source satellites appear in ``auth_sources``.

- `**has_self**`: ``True`` when the destination satellite also appears among the sources.
- `**nav_authenticated**`: ``True`` when the navigation data of ``prn_d`` is authenticated in that subframe.

4) Skyplot

The skyplot is built from UBX NAV-SAT messages in `navsat.csv`, parsed by `extract_navsat.py` and plotted in `plot_results.py`.

It will include any Galileo satellite (`gnssId == 2`) that the receiver reported in NAV-SAT (even if you never decoded/accepted OSNMA pages from it).

There could be some Galileo satellites that appear in the skyplot but not in the timelines (e.g., not in the `page_log.csv`), those are the reasons:

- The UBX stream contains NAV-SAT for that satellite, but no RXM-SFRBX E1B pages for it (or they were not recorded).
- RXM-SFRBX pages exist but are filtered out before logging (CRC fail, dummy/alert pages, non-E1B, etc.) by `_filter_page()` in `receiver.py`.
- The runner itself may pre-filter what goes into the processed UBX (e.g., `run_file.py` writes a reduced `gal_e1b.ubx` that keeps NAV-SAT but only RXM-SFRBX with Galileo E1B).

5) NAV-SAT Parameter Timelines

These three heatmaps are generated from ``navsat.csv`` and show, for each Galileo satellite visible during the run, how `**C/NO**`, `**elevation**`, and `**azimuth**` evolve over time. They use the same continuous 2-second time grid as the OSNMA authentication timelines, so you can visually align signal quality with authentication results.

Each plot is a matrix where:

- `**Y-axis**`: Galileo SVID (one row per satellite)
- `**X-axis**`: absolute time in 2-second bins (same ``(wn, tow)`` grid as ``page_log.csv``)
- `**Color**`: the value of the plotted parameter at that moment for that satellite

White cells mean no NAV-SAT report was received for that satellite at that time bin.

C/NO Timeline

Shows the carrier-to-noise density ratio in ****dB-Hz****. It is a direct indicator of signal quality: high values (yellow, ~60 dB-Hz) indicate a strong, clean signal; low values (dark purple, near 0) indicate a weak or heavily attenuated signal. Sudden drops to dark purple mid-track (as visible for E18 and E25 in the example) typically correspond to signal blockage or multipath events. Satellites that appear only at the edges of the timeline (e.g., E08, E09) were rising or setting during the recording.

Elevation Timeline

Shows the satellite elevation angle in ****degrees**** (0° = horizon, 90° = zenith). This plot is useful to interpret both the C/N0 and the authentication results: low-elevation satellites tend to have weaker signals and are more prone to multipath, which can cause OSNMA pages to be discarded by `\_filter\_page()` before they even reach the authentication pipeline.

Azimuth Timeline

Shows the satellite azimuth angle in ****degrees**** (0°/360° = North, 90° = East). Useful for understanding the geometric diversity of the constellation during the run and for diagnosing obstructions: a satellite with constant azimuth pointing toward a known obstacle (e.g., a building) will often show a degraded C/N0 in the first plot.

PDOP Timeline

Shows the Position Dilution of Precision (PDOP) in ****dimensionless units****. PDOP is a measure of the geometric quality of the satellite constellation for positioning: lower values indicate better geometry and more accurate position estimates. PDOP is computed from the receiver's estimated position and the visible satellites at each 2-second bin.

NAV-SAT messages carry an `iTOW` in milliseconds relative to the current GPS week. To align them with the absolute time grid derived from `page\_log.csv`, the code applies a ±604800 s correction when the raw `iTOW`-derived timestamp falls more than ~3.5 days outside the visible window. This means:

- The recorded UBX file must not span more than ****3.5 days****.
- There must be no data gap longer than ****3.5 days**** within the file.

If either condition is violated, some NAV-SAT rows may be placed in the wrong time bin or discarded entirely.

6) Authenticated Nav Data vs Complete Nav Data vs Missing Authentication vs Total Satellites in LOS Timeline

This line chart is generated from `status\_log.json` and provides a ****per-subframe count**** of satellites in two distinct states:

- ****Total Satellites tracked****: satellites which are being tracked by the receiver and their physical parameters are reported.

- ****Authenticated nav data****: satellites whose navigation data (clock + ephemeris) has been verified by the OSNMA protocol in that subframe. These are the satellites that appear in ``nav_authenticated`` in ``tag_sources_by_subframe.csv``.
- ****Complete nav data****: satellites that have a full, usable Ephemeris Data (CED) block available extracted from this subframe. These are candidates for authentication in upcoming subframes, provided their tags arrive and verify correctly.
- ****Missing Authentication - Previous subframe's complete Nav Data not Authenticated****: Satellites in this current subframe that successfully provided complete navigation data in the previous subframe, but failed to achieve an authenticated state in this subframe's period.

What one point represents

Each point on the X-axis corresponds to the start TOW of one 30-second GST subframe present in ``status_log.json``. Unlike the heatmaps in Section 5, this plot does ****not**** use a continuous 2-second grid: only subframes that the pipeline actually processed and logged are shown. Gaps in the X-axis therefore reflect either missing subframes or subframes that were discarded before reaching the status log.

X-axis label format

- If all subframes belong to a ****single GST week****, the X-axis shows only the subframe TOW in seconds.
- If subframes span ****multiple GST weeks****, the label switches to ``wn:tow`` format to avoid ambiguity.

Bibliography

- [1] Thiago Azevedo de Vasconcelos, Lothar Kurz, Andriy Konovaltsev, Tobias Ehlers, and Michael Meurer. «Satellite Navigation Interference Monitoring in the Baltic and North Seas». In: *The International Hydrographic Review* 30.2 (2024), pp. 82–97 (cit. on p. 1).
- [2] Mark L. Psiaki and Todd E. Humphreys. «GNSS Spoofing and Detection». In: *Proceedings of the IEEE* 104.6 (2016), pp. 1258–1270. DOI: 10.1109/JPROC.2016.2526658 (cit. on pp. 1, 14).
- [3] Fabio Dovis, ed. *GNSS Interference Threats and Countermeasures*. GNSS Technology and Applications Series. Norwood, MA: Artech House, 2015. ISBN: 9781608078103 (cit. on pp. 1, 13, 14).
- [4] *Galileo OSNMA SDD*. July 2025 (cit. on pp. 1, 5, 17, 18, 21, 23, 31).
- [5] *Galileo OSNMA Receiver Guidelines*. Guidelines. European Union, Jan. 2024 (cit. on pp. 1, 14, 18, 21, 24, 25, 27–29, 31–33).
- [6] Daniele Borio, Fabio Dovis, Heidi Kuusniemi, and Letizia Lo Presti. «Impact and Detection of GNSS Jammers on Consumer Grade Satellite Navigation Receivers». In: *Proceedings of the IEEE* 104.6 (2016), pp. 1233–1245. DOI: 10.1109/JPROC.2016.2543266 (cit. on pp. 2, 14, 16).
- [7] Dennis M. Akos. «Who’s Afraid of the Spoofer? GPS/GNSS Spoofing Detection via Automatic Gain Control (AGC)». In: *NAVIGATION: Journal of the Institute of Navigation* 59.4 (2012), pp. 281–290 (cit. on pp. 2, 16).
- [8] Kyle D. Wesson, Jason N. Gross, Todd E. Humphreys, and Brian L. Evans. «GNSS Signal Authentication via Power and Distortion Monitoring». In: *IEEE Transactions on Aerospace and Electronic Systems* 54.2 (2018), pp. 739–754. DOI: 10.1109/TAES.2017.2765258 (cit. on pp. 2, 16).
- [9] Paul Y. Montgomery, Todd E. Humphreys, and Brent M. Ledvina. «Receiver-Autonomous Spoofing Detection: Experimental Results of a Multi-Antenna Receiver Defense Against a Portable Civil GPS Spoofer». In: *Proceedings of the 2009 International Technical Meeting of the Institute of Navigation (ION ITM 2009)*. Anaheim, CA, 2009, pp. 124–130 (cit. on pp. 2, 16).

- [10] W. De Wilde J. - M. Sleewaegen, B. Bougard, G. Cuypers, S. A. Popugaev, et al. «Authentication by polarization: a powerful anti-spoofing method». In: (Sept. 2018) (cit. on pp. 2, 16).
- [11] Parisa Borhani Darian and Pau Closas. «Deep Neural Network Approach to GNSS Signal Acquisition». In: 2020, pp. 1214–1223. DOI: 10.1109/PLANS46316.2020.9110205 (cit. on pp. 2, 70).
- [12] Akmal Rustamov, Alex Minetto, and Fabio Dovis. «Improving GNSS Spoofing Awareness in Smartphones via Statistical Processing of Raw Measurements». In: *IEEE Open Journal of the Communications Society* 4 (2023), pp. 873–891. URL: <https://ieeexplore.ieee.org/document/10081330> (cit. on pp. 2, 14).
- [13] Y. Wang, X. Chen, D. Zou, et al. «Incremental learning-enhanced LSTM for spoofing detection in GNSS signals». In: *GPS Solut* 30 (2026), p. 32. DOI: 10.1007/s10291-025-01996-5 (cit. on pp. 2, 70).
- [14] Roberto Calvo-Palomino, Arani Bhattacharya, Gerome Bovet, and Domenico Giustiniano. «Short: LSTM-based GNSS Spoofing Detection Using Low-cost Spectrum Sensors». In: 2020, pp. 273–276. DOI: 10.1109/WoWMoM49955.2020.00055 (cit. on pp. 2, 70).
- [15] Joseph Igot. *GPS Spoofing Sophistication Real-Time Characterization Framework via Hybrid Unsupervised Model on Receiver Observables*. Nov. 2025. DOI: 10.36227/techrxiv.176308141.16159170/v1 (cit. on pp. 2, 71).
- [16] S. Kocher, D. Contreras Franco, A. Dietz, and A. Rügamer. «On the Edge Model-Aided Machine Learning GNSS Interference Classification with Low-Cost COTS Hardware». In: *Engineering Proceedings* 88.1 (2025), p. 51. DOI: 10.3390/engproc2025088051 (cit. on pp. 2, 71).
- [17] Logan Scott. «Anti-Spoofing and Authenticated Signal Architectures for Civil Navigation Systems». In: *Proceedings of the 16th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS/GNSS 2003)*. Portland, OR, 2003, pp. 1543–1552 (cit. on pp. 2, 17).
- [18] Ignacio Fernández-Hernández, Vincent Rijmen, Gonzalo Seco-Granados, Javier Simón, Irma Rodríguez, and J. David Calle. «A Navigation Message Authentication Proposal for the Galileo Open Service». In: *NAVIGATION: Journal of the Institute of Navigation* 63.1 (2016), pp. 85–102. DOI: 10.1002/navi.125 (cit. on pp. 2, 17).
- [19] Todd E. Humphreys. «Detection Strategy for Cryptographic GNSS Anti-Spoofing». In: *IEEE Transactions on Aerospace and Electronic Systems* 49.2 (2013), pp. 1073–1090 (cit. on pp. 2, 17).

- [20] Galan, Aleix and Fernandez-Hernandez, Ignacio and Cucchi, Luca and Seco-Granados, Gonzalo. «OSNMALib: An Open Python Library for Galileo OS-NMA». In: Apr. 2022, DOI: {10.1109/NAVITEC53682.2022.9847548} (cit. on pp. 2, 35).
- [21] Hirokawa, Rui and Hauschild, André and Everett, Tim. «Python Toolkit for Open PPP/PPP-RTK Services». In: Oct. 2023, 2514–2526. DOI: {10.33012/2023.19207} (cit. on pp. 2, 34).
- [22] J.M. Juan Zornoza J. Sanz Subirana and M. Hernández-Pajares. *GNSS DATA PROCESSING, VOLUME I: Fundamentals and Algorithms*. Ed. by K. Fletcher. Vol. I. ESA Communications, May 2013 (cit. on pp. 5–8, 10).
- [23] *GPS Constellation Status*. Accessed: 2026-07-08. Navigation Center, United States Coast Guard (USCG). URL: <https://www.navcen.uscg.gov/gps-constellation> (cit. on p. 5).
- [24] *Galileo Constellation Information*. Accessed: 2026-07-08. European GNSS Service Centre (GSC). URL: <https://www.gsc-europa.eu/system-service-status/constellation-information> (cit. on p. 5).
- [25] *Galileo OS SIS ICD*. Nov. 2025 (cit. on pp. 6, 7, 21).
- [26] *Galileo OS SDD*. Nov. 2023 (cit. on pp. 6, 18).
- [27] *Code Based Positioning (SPS)*. Navipedia, European Space Agency (ESA). URL: [https://gssc.esa.int/navipedia/index.php?title=Code_Based_Positioning_\(SPS\)](https://gssc.esa.int/navipedia/index.php?title=Code_Based_Positioning_(SPS)) (cit. on p. 9).
- [28] Department of Homeland Security (DHS) Science and Technology Directorate. *Resilient Positioning, Navigation, and Timing (PNT) Conformance Framework*. Tech. rep. Version Version 2.0. Washington, D.C., USA: U.S. Department of Homeland Security, 2023 (cit. on p. 13).
- [29] Beatrice Motella, Marco Pini, and Fabio Dovis. «Investigation on the effect of strong out-of-band signals on Global Navigation Satellite Systems receivers». In: *GPS Solutions* 12.2 (2008), pp. 77–86. DOI: 10.1007/s10291-007-0085-5 (cit. on p. 13).
- [30] Maxandre Coulon, Alexandre Chabory, Axel Garcia-Pena, Jérémy Vezinet, Christophe Macabiau, Philippe Estival, Pierre Ladoux, and Benoît Roturier. «Characterization of Meaconing and its Impact on GNSS Receivers». In: *Proceedings of the 33rd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2020)*. 2020, pp. 3713–3737. DOI: 10.33012/2020.17713 (cit. on p. 14).

- [31] Todd E. Humphreys, Brent M. Ledvina, Mark L. Psiaki, Brady W. O'Hanlon, and Paul M. Kintner Jr. «Assessing the Spoofing Threat: Development of a Portable GPS Civilian Spoofer». In: *Proceedings of the 21st International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2008)*. Savannah, GA, 2008, pp. 2314–2325 (cit. on p. 14).
- [32] Rigas T. Ioannides, Thomas Pany, and Glen Gibbons. «Known Vulnerabilities of Global Navigation Satellite Systems, Status, and Potential Mitigation Techniques». In: *Proceedings of the IEEE* 104.6 (2016), pp. 1174–1194 (cit. on p. 15).
- [33] Ali Jafarnia-Jahromi, Ali Broumandan, John Nielsen, and Gerard Lachapelle. «GPS Vulnerability to Spoofing Threats and a Review of Antispoofing Techniques». In: *International Journal of Navigation and Observation* 2012 (2012), p. 127072. DOI: 10.1155/2012/127072 (cit. on p. 15).
- [34] Elliott D. Kaplan and Christopher J. Hegarty, eds. *Understanding GPS/GNSS: Principles and Applications*. 3rd ed. Norwood, MA: Artech House, 2017 (cit. on p. 16).
- [35] Samer Khanafseh, Naeem Roshan, Steven Langel, Fang-Cheng Chan, Mathieu Joerger, and Boris Pervan. «GPS Spoofing Detection Using RAIM with INS Coupling». In: *Proceedings of the IEEE/ION Position, Location and Navigation Symposium (PLANS 2014)*. Monterey, CA, 2014, pp. 1232–1239 (cit. on p. 16).
- [36] Micaela Troglia Gamba, Minh Duc Truong, Beatrice Motella, Emanuela Falletti, and Tung Hai Ta. «Hypothesis testing methods to detect spoofing attacks: a test against the TEXTBAT datasets». In: *GPS Solutions* 21.2 (2017), pp. 577–589 (cit. on p. 16).
- [37] Sagar Dasgupta, Kazi Hassan Shakib, and Mizanur Rahman. *Experimental Validation of Sensor Fusion-based GNSS Spoofing Attack Detection Framework for Autonomous Vehicles*. 2024. URL: <https://arxiv.org/abs/2401.01304> (cit. on pp. 17, 70, 71).
- [38] S. Taghizadeh and R. Safabakhsh. «An integrated INS/GNSS system with an attention-based hierarchical LSTM during GNSS outage». In: *GPS Solut* 27 (2023), p. 71. DOI: 10.1007/s10291-023-01412-w (cit. on pp. 17, 70).
- [39] *Galileo OSNMA SIS ICD*. Interface Control Document. European Union. Oct. 2023 (cit. on pp. 18, 21–28, 31).
- [40] Adrian Perrig, Dawn Song, Ran Canetti, J. D. Tygar, and Bob Briscoe. *Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction*. RFC 4082. June 2005. URL: <https://www.rfc-editor.org/rfc/rfc4082> (cit. on pp. 18, 25, 27, 32).

- [41] Jason Anderson. *Authentication Security of PRF GNSS Ranging*. arXiv preprint arXiv:2510.02196. 2025. URL: <https://arxiv.org/abs/2510.02196> (cit. on p. 18).
- [42] Tara Mina, Ashwin Kanhere, Akshay Shetty, and Grace Xingxin Gao. «GPS Spoofing-Resilient Filtering Using Self-Contained Sensors and Chimera Signal Enhancement». In: *NAVIGATION: Journal of the Institute of Navigation* 71.1 (2024). DOI: 10.33012/navi.621 (cit. on p. 18).
- [43] *PKI System CP/CPS*. Certificate Policy and Certification Practice Statement. Jan. 2024 (cit. on pp. 23, 27).
- [44] *Galileo OSNMA IDD ICD*. Interface Control Document. European Union, Jan. 2024 (cit. on p. 27).
- [45] Aiman Al-Sabaawi. «Cryptanalysis of Block Cipher: Method Implementation». In: *2022 International Conference for Advancement in Technology (ICONAT)*. 2022, pp. 1–7. DOI: 10.1109/ICONAT53423.2022.9726054 (cit. on p. 34).
- [46] Galan, Aleix and Iñiguez, Cristian and Fernandez-Hernandez, Ignacio and Pollin, Sofie and Seco-Granados, Gonzalo. «Improving OSNMAlib: New Formats, Features, and Monitoring Capabilities». In: *IEEE Journal of Indoor and Seamless Positioning and Navigation* 3 (Jan. 2025), 117–127. DOI: {10.1109/JISPIN.2025.3558771} (cit. on p. 36).
- [47] Galan, Aleix and Fernandez-Hernandez, Ignacio and Wilde, Wim and Pollin, Sofie and Seco-Granados, Gonzalo. «Improving Galileo OSNMA Time to First Authenticated Fix». In: *IEEE Transactions on Aerospace and Electronic Systems* 61 (Jan. 2025), 11720–11735. DOI: {10.1109/TAES.2025.3570273} (cit. on p. 36).
- [48] *RINEX The Receiver Independent Exchange Format*. July 2015 (cit. on p. 36).
- [49] Norwegian Communications Authority (Nkom) and Norwegian Defence Research Establishment (FFI). *Jammertest 2025: Test Catalogue and Transmission Plan*. Official GNSS Interference Test Campaign Documentation, Bleik, Andøya, Norway. Campaign schedule and transmission ground truth. Sept. 2025 (cit. on pp. 69, 75, 82).
- [50] A. Enan, M. Chowdhury, S. Dasgupta, and M. Rahman. *Quantum-Classical Hybrid Framework for Zero-Day Time-Push GNSS Spoofing Detection*. 2025. URL: https://ui.adsabs.harvard.edu/link_gateway/2025arXiv250818085E/arxiv:2508.18085 (cit. on pp. 70, 71).
- [51] Nishant S. Gaikwad, Lucas Heublein, Nisha L. Raichur, Tobias Feigl, Christopher Mutschler, and Felix Ott. *Federated Learning with MMD-based Early Stopping for Adaptive GNSS Interference Classification*. 2024. URL: <https://arxiv.org/abs/2410.15681> (cit. on p. 71).

- [52] Peng Wu, Helena Calatrava, Tales Imbiriba, and Pau Closas. «Federated Learning of Jamming Classifiers: From Global to Personalized Models». In: *NAVIGATION: Journal of the Institute of Navigation* (Mar. 2025). DOI: 10.33012/navi.688 (cit. on p. 71).
- [53] Harshith Manjunath, Lucas Heublein, Tobias Feigl, and Felix Ott. «Multimodal+to-Text Prompt Engineering in Large Language Models Using Feature Embeddings for GNSS Interference Characterization». In: Mar. 2025, pp. 1–7. DOI: 10.1109/WCNC61545.2025.10978760 (cit. on p. 71).