A Spoofing Detection Method for Civilian L1 GPS and the E1-B Galileo Safety of Life Service

Chris J. Wullems, *Qascom S.r.l.*

***Abstract*—This paper describes an effective method for signal-authentication and spoofing detection for civilian GNSS receivers using the GPS L1 C/A and the Galileo E1-B Safety of Life service. The paper discusses various spoofing attack profiles and how the proposed method is able to detect these attacks. This method is relatively low-cost and can be suitable for numerous mass-market applications. This paper is the subject of a pending patent.**

***Index Terms*—Spoofing Detection, Global Positioning System, Galileo, Signal Authentication, Security**

I. Introduction

In recent years, Global Satellite Navigation Systems (GNSS) have seen near exponential growth in the number of low-cost precise timing and positioning applications in the market. The acceptance of this technology by many industries has led to its widespread adoption, often without consideration of the potential security risks the use of GNSS can have on safety and financially critical applications.

Trusted positioning and timing services are not only a pre-requisite for safety and financially critical applications, but also a requirement for location-based security services such as geo-encryption and position attestation. To date, location-based security services using GNSS have had limited success for mass-market applications, due to the lack of suitable low-cost spoofing-detection and anti-spoofing technologies.

Spoofing and jamming pose serious threats to such applications. While a jamming attack typically aims to disrupt or degrade the performance of a GNSS receiver, by transmitting radio signals that interfere with genuine GNSS signals received from space, a successful spoofing attack can be significantly more hazardous. Spoofing aims to deceive a GNSS receiver as to its time and position by generating a simulated GNSS signal that appears to be genuine to the receiver.

Attack scenarios for applications such as geo-fencing (e.g. vessel monitoring for fish stock management) and GNSS-based digital rights management differ in that an attacker would attempt to spoof the position / time of his own receiver in order to defeat a GNSS-based access control system (self-spoofing). As the attacker is in possession of the GNSS receiver, a GNSS simulator could be connected directly to the receiver using an RF cable before it is turned on. Spoofing under these conditions is more difficult to detect if the simulation is accurate, as no genuine GNSS signals are visible to the receiver.

The cost of orchestrating a spoofing attack is no longer a determining risk factor, as signal simulators can be rented cheaply and sophisticated spoofing attacks can be developed on low-cost software-defined radio platforms that are readily available. Such platforms provide significant open-source software support. In this paper we propose a technique for the detection of spoofing on open civilian GNSS signals, which is suitable for low-cost GNSS receivers. The proposed technique addresses the spoofing scenarios that are discussed in this paper.

Manuscript received 31/10/2010.

C. J. Wullems investigated spoofing detection at Qascom S.r.l., Bassano del Grappa, VI, 3606 Italy (phone: +39-0424-525-473; fax: +39-0424-527-800; e-mail: chris.wullems@qascom.it).

II. Background

This section provides a high-level overview of the GNSS receiver functions relevant to the spoofing detection method. Figure 1 illustrates the fundamental components of a typical GNSS receiver: a Low-Noise Amplifier (LNA), an RF Front-End and a baseband processor. The RF front end is responsible for the down-conversion of the signal to an intermediate frequency and the digitization of the signal using an analogue to digital converter (ADC). The baseband processor acquires and tracks satellites using the digitized signal.

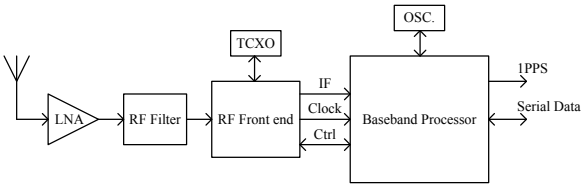

Figure 1. Basic GNSS receiver

Figure 2 illustrates the logical processing blocks in the baseband processor. The acquisition function involves finding the carrier frequency and code phase for each visible satellite in the digitized signal. This information is passed to the tracking function, which is responsible for keeping track of the code phase of each acquired satellite in the signal. In order to track and demodulate the signal of a satellite, the tracking module has to generate two replicas, one for the carrier and one for the code. To produce the exact replicas, a carrier tracking loop (Costas loop) and a code tracking loop (delay lock loop) are required.

A delay lock loop (DLL) is a tracking loop that correlates the input signal with three replicas of the code (early, prompt and late), each nominally generated with a typical spacing of ± ½ a chip. The three outputs are integrated and dumped, providing an indication as to how much the specific code replica correlates with the code in the incoming signal, and therefore whether the signal is in phase, early or late by ½ a chip. If the signal is not in phase, the replica is then adjusted so that it is in phase. These functions are typically performed in hardware.

The navigation data extraction function performs bit and frame synchronization using the code phase information provided by the tracking module, such that the beginning of a subframe can be identified and the navigation data can be obtained. From the identified beginning of subframes for a set of channels, the pseudoranges can be computed. The navigation data provides the necessary data for applying clock corrections, calculating the positions of the satellites, and correcting the pseudoranges for tropospheric and ionospheric delays.

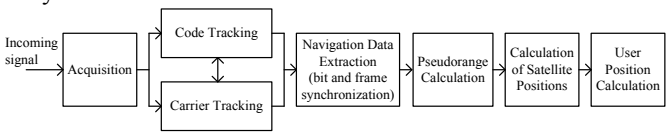

Figure 2. Logical processing blocks in the baseband processor

The user position is calculated by solving a nonlinear equation used to determine the user position and clock bias from the satellite positions and corrected pseudorange observations for at least four satellites, commonly calculated by linearizing the equation using the least-squares method. The pseudorange calculation process of the GNSS receiver is of particular interest to the spoofing detection method presented in this paper. The following subsection briefly describes this function.

A. Pseudorange calculation

Pseudorange measurements are computed as the travel time (τ_i^k) of the GNSS signal from satellite k to receiver i. The relationship between τ_i^k and the pseudorange (P_i^k) is defined in Equation (1.1) as:

$$t_i - t^k = \tau_i^k = P_i^k / c \qquad (1.1)$$

where, t_i is the time at which the GNSS signal was received by receiver i, t^k is the time of transmission of the GNSS signal by satellite k, and c is the velocity of light in a vacuum.

In obtaining the digitized signal data, there is no absolute time reference and the only time reference is the sampling frequency. As a result, pseudoranges can be measured only in a relative way, as the time difference between the start of a subframe with respect to a reference satellite (the satellite with the earliest arriving subframe). The start of a subframe is identified by correlating the navigation data with the preamble. This provides the frame (code epoch / millisecond of the signal for GPS) in which the subframe starts. Subsequent steps must be taken to validate the navigation data, by verifying parity, etc.

In order to obtain a pseudorange with sufficient precision, the start of the spreading code in the specific frame must be found. The resolution of the pseudorange measurement is the sample frequency. The number of samples contained in a frame is a multiple of 1023 for GPS or 4092 for Galileo E1-B, where a frame represents the code epoch (1ms for GPS or 4ms for Galileo E1-B).

III. SPOOFING SCENARIOS

This subsection discusses a number of likely spoofing scenarios for both GPS and Galileo, given the target and application domain of low-cost GNSS receivers. Spoofing attacks such as Meaconing, involving the delay and rebroadcast of GNSS signals, are not discussed in this paper. Such methods do not involve simulation, but rather the delay of genuine GNSS signals in order to create large errors in the navigation solution. Such techniques involve very low cost equipment and can be assembled from readily available components using schematics found on the Internet. One such system [1] claims to be able to spoof the position of a GPS receiver approximately 100 meters (300 feet) . Meaconing also affects encrypted signals such as the GPS P(Y) signal and the Galileo Public Regulated Service (PRS).

The spoofing scenarios that will be discussed in this paper involve simulation of GNSS signals. The primary objective of this type of spoofing attack is to deceive the target receiver by simulating GNSS signals in order to provide the target receiver with a falsified position and/or time. For the purpose of this paper, we group simulation-based spoofing attacks into the following three categories:

- **Unsynchronized:** We define an unsynchronized spoofing as an attack that involves radiating the signal from a GNSS simulator towards the target receiver using an antenna or RF cable, where the simulator is autonomous (i.e. not synchronized with the GNSS constellation). The 1PPS (1 Pulse Per Second) output of an external GNSS receiver may be used to synchronize the time of the simulation to GNSS. This type of attack is also characterized as an unsophisticated attack as it can be accomplished with off-the-shelf equipment and minimal configuration.
- **Loosely Synchronized:** We define a loosely synchronized spoofing as an attack that aims to mimic the actual constellation in terms of time, Doppler frequency, pseudorange, navigation data and C/N0 for each channel. This type of attack can be developed with a trivial amount of software development and off the shelf equipment including a simulator supporting remote control of signal generation parameters, and a GNSS receiver that provides the required raw measurements. This type of attack does not achieve synchronization of the simulation and the actual constellation to within half a chip of the actual GNSS signal for each channel (in order to be captured by the early, late or prompt correlator). The attack will most probably require jamming or significantly higher signal power in order to force the receiver to lose code lock and initiate re-acquisition. The loss of code lock and bit / frame synchronization are potentially indicators of spoofing or tracking in a difficult environment (e.g. urban canyon or entry in a tunnel). A spoofer may

attempt to hide such evidence by initiating an attack in a tunnel or difficult environment, or by creating in-band interference that significantly degrades navigation performance.

- **Tightly Synchronized:** We define tightly synchronized spoofing as an attack that attempts to synchronize the simulated signal to within half a chip of the actual signal, such that the receiver does not loose code lock and will start tracking the spoofed signal due to the higher correlation value of the spoofed signal. This type of attack most likely requires purpose built hardware, as it relies on functionality not commonly found in commercial simulators. It is important that the power of the spoofed signal is not significantly higher than the actual signal and consistently higher across all channels as to raise suspicion. By facilitating the capture of each channel in a non-obvious way (i.e. a single channel at a time, random channel selection and random intervals between each channel capture), changes in the observed C/N0 should not raise suspicion.

The following subsections discuss previous work in relation to the types of spoofing attacks described above and provide a discussion of loosely and tightly synchronized spoofing scenarios. The unsynchronized scenario is not discussed in this paper, as this type of spoofing can be easily detected by a number of simple signal sanity checks as detailed in Section III.A.

A. Previous Work

The Volpe report [2] noted that in 2001 there were no practical mitigation methods available for spoofing attacks, and that a few potentially effective techniques would be too expensive for civilian applications, in particular intelligent transportation systems. Based on the literature reviewed, we believe that this is still the case. Approaches to providing an anti-spoofing capability for civilian receivers include navigation message authentication; the insertion of non-deterministic hidden markers or spread spectrum security codes in the signal; exploitation of signals that civilians do not have access to (e.g. the P(Y) of GPS) and are unable to simulate; and simple consistency checks.

Navigation message authentication (NMA) has been proposed by numerous researchers [3] [4] [5] and involves the reception of authentication data by a GNSS receiver (either within the navigation message or provided by a third-party), allowing it to authenticate the source of the navigation message and verify its cryptographic integrity (i.e. detection of unauthorized modification of the message). Conceptually, an adversary would not be able to simulate authentication data, as he/she would not have the keys required to generate it. The spoofing protection afforded by NMA is minimized by the fact that a spoofer can acquire the legitimate signal and replay the navigation message (containing the authentication data) over the simulated signal.

Spread spectrum security codes or hidden markers have been proposed by Scott in [3] and Kuhn in [5]. Spectrum Security Codes (SSSC), interleaved with the normal spreading codes, are used to facilitate signal authentication. These codes or hidden markers are not perceivable by an adversary, as the power of signal received from space is below thermal noise. A receiver is able to authenticate the signal on receipt of an authentication message (transmitted in the navigation message) by a receiver, which is used to generate a replica of the SSSC sequence. The message is released several minutes after the sequence has already been transmitted. This allows the receiver to de-spread previously collected and stored samples. The signal is authenticated when the SSSC is detected at the correct power level. Both NMA and SSSCs require modifications to GPS / Galileo messages and/or signal design. SSSCs in particular can result in deteriorated navigation performance, and have therefore not been considered in evolutions of GPS and Galileo.

Other methods for anti-spoofing include signal access control achieved through encryption of the spreading code. Such mechanisms typically involve the modolo-2 sum of the spreading sequence with a encrypting code. An example of this is the P(Y) signal of GPS. The P(Y)-code is the modulo-2 sum of the P-code and an unknown encrypting code called the W-code. The W-code is a pseudo-random sequence of chips that occur at a rate of 511.5 kHz, such that there are 20 P-code chips for every W-code chip. The Y-code cannot be de-spread by a replica P-code unless it is decrypted. Decryption consists of multiplying the Y-code by a receiver-generated replica of the W-code made available only to authorized users. Since the encrypting W-code is not known by spoofers, illegitimate signals can be easily identified as spoofers cannot recreate the P(Y) signal. Unfortunately, the P(Y) signal is not available to civilian users.

An anti-spoofing method proposed by [6] uses the P(Y) signal to authenticate GPS signals for civilian receivers. The method involves the use of a network of reference stations with high-gain antennas to provide raw signal samples of P(Y) signals. The GPS receiver also provides raw signal samples to an authentication server, that once corrected for Doppler, are correlated with the signal samples obtained from the reference stations. The known phase relationship of the P(Y) to the C/A is fundamental in the authentication scheme, allowing Doppler and the start of a given set of samples to be identified. The high-gain antennas are necessary to increase the signal to noise ratio, as the P(Y) is immersed in thermal noise, and multiplying the two sets of signal samples also multiplies the noise. In order to improve the performance of the method proposed by [6], additional processing gain could be obtained by multiplying the signal by the known P-code in order to recover the W-code. As the P(Y)-code is a modulo-2 sum of the P-code and W-code at a rate of 511.5 kHz, the P-code can be removed using a locally generated replica. If successfully removed, only the W-code modulation should remain. This should effectively reduce the bandwidth from about 20 MHz to about 1 MHz, providing significant processing gain.

Although the above method would be very effective in authenticating GPS signals, this is definitely not a mass-market approach. Not only is there significant cost for receivers in terms of front-ends able to down-convert and sample signals of a bandwidth 10 times that of the C/A, the infrastructure required to facilitate the authentication (the network of ground stations) also represents a significant cost. This paper focuses on low-cost anti-spoofing methods suitable for mass-market receivers.

Warner et. al. in [7] presents a number of low-cost GPS spoofing countermeasures, some of which can be applied as a software upgrade and others that involve retrofitting the GPS receiver. The first three methods described by the authors are based on monitoring the absolute GPS signal strength, changes in the signal strength over time and the variance of signal strengths for each satellite signal for unrealistic values from an unsophisticated spoofer. Such methods assume the naive use of a GPS constellation simulator for spoofing.

Other countermeasures proposed by Warner et. al. in [7] are based on recognizing characteristics of the simulator, such as monitoring the PRNs and satellite geometry in the case a spoofer is not attempting to mimic the true satellite constellation, monitoring the times at which various satellites appear based on the assumption that all satellites will be immediately visible in a non-sophisticated spoofing attack; and monitoring the GPS time with respect to a local clock. Most of these countermeasures can be easily defeated by loosely synchronizing the simulation.

Wen et. al in [8] proposes a number of similar countermeasures for single frequency GPS receivers including: monitoring of signal power (absolute, rate of change and relative signal strength between carriers); validation of Doppler shift; validation of satellite positions and ephemeris; and jump detection. They additionally propose methods for dual frequency receivers including cross-correlation of P(Y) on L1 and L2 and monitoring of range differences between L1 and L2 for differences caused by delay in the ionosphere. The dual frequency methods are not suitable for a mass-market receivers for the reasons articulated above.

One of the low-cost anti-spoofing defenses proposed by Humphreys et. al. in [9] involves monitoring consecutive accumulations at the C/A code-length interval for unexpected sign changes. As the mechanisms they propose are autonomous (i.e. they do not connect to an external observer to authenticate the data), for spoofing to be detected, the receiver must have been tracking legitimate GNSS signals prior to spoofing, or there must be the presence of at least one legitimate PRN in order for the difference in accumulated C/A code phases to be observed. Figure 3 illustrates this concept with the GPS L1 C/A bit train.

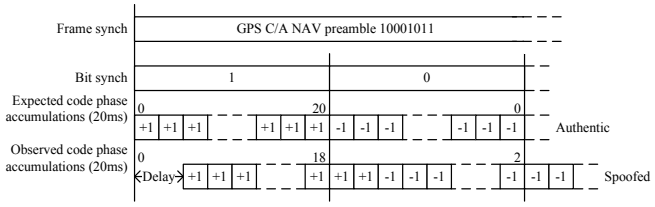

Figure 3. Expected and observed code phase accumulations for GPS C/A NAV under spoofing conditions

A limitation of this method is that low signal power or unintentional interference could cause unexpected sign changes thereby raising false spoofing alarms. This method would not detect spoofing scenarios in which the receiver is powered on in the presence of spoofing, such that the spoofer is connected directly to the receiver via an RF cable and legitimate GNSS signals are not visible to the receiver. In this case, differences would not be observed in code phase accumulations. The following sections describe loosely and tightly synchronized spoofing scenarios.

B. Loosely synchronized spoofing scenario

In this scenario, a loosely synchronized spoofing attack would involve the synchronization of an off-the-shelf simulator with a GNSS receiver, using a PC for the generation of predicted navigation data and for the remote control of the simulator (Figure 4).

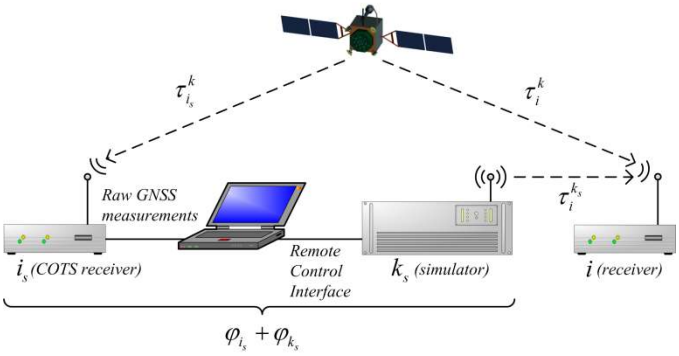

Figure 4. Loosely synchronized spoofing scenario

This type of spoofing presents a potentially serious risk to GNSS applications deployed in safety and financially critical applications. Although 12-channel constellation simulators are still very expensive to acquire, they can be rented cheaply and the technical / engineering expertise required to perform a loosely synchronized spoofing attack is negligible.

A number of mass-market receivers support binary protocols that provide access to raw measurements including the following:

- Raw GPS subframes or Galileo words transmitted asynchronously within milliseconds of the parity verification;
- Raw pseudoranges;
- Accumulated carrier cycles for each channel;
- Doppler frequency for each channel; and
- C/N0 in dBHz for each channel

Such raw measurements can be used to remotely control the simulation, such that the channel Doppler frequency, pseudoranges and signal power for each channel are consistent with those observed by the receiver. The desired position can then be obtained by ramping up or down the pseudoranges via the simulator's remote command interface.

The raw GPS subframes / Galileo words provide the data bits required to replicate the navigation message, such that once sufficient bits of the new data set are collected, the navigation message can be predicted (i.e. TOW, parity, etc. updated).

In order to synchronize the time of the simulation, the 1PPS output of the GNSS receiver can be used. If the absolute signal strength is an issue, a hardware attenuator can be used to limit the signal power from the simulator, such that it is within the expected GPS signal power threshold (about -160 dBW [10]).

The following subsections discuss various methods of replicating navigation data, such that the navigation data modulated on the simulated signal is consistent with the navigation data observed from the actual GNSS.

1) Navigation message prediction

Humpheries et. al. [9] discusses the design of a GPS receiver / spoofer in which the delay incurred by relaying the data bits from the GPS receiver is removed by predicting data bits given knowledge of the structure and recent bit observations. As the authors note, there are issues with this method near the two-hour ephemeris update boundaries, where it would not be possible to accurately predict the data.

For the GPS L1 navigation message, new datasets containing ephemeris and clock parameters are transmitted every two hours during normal operation, with a corresponding curve-fit interval of four hours [10]. The minimum amount of time required to obtain the entire navigation data set (all pages of subframes 4 and 5) is 12.5 minutes, assuming there are no issues demodulating the navigation data. After a data cut-over boundary, a spoofer would have to wait for at least 18 seconds for the first three subframes containing clock parameters and ephemerides before being able to predict navigation message, assuming the spoofer was able to obtain the other data sets that are updated less frequently beforehand.

A page within the navigation message of particular interest is the Navigation Message Correction Table (NMCT), transmitted in subframe 4. This page contains non-predictable data bits which would require the spoofer to wait a further 6.5 minutes if the spoofer does not want to introduce a delay in the simulated data stream (assuming subframe 4 transmission starts from page 1 for a new data set). The NMCT page is typically updated in intervals consistent with ephemeris updates. Navigation data prediction should be able to defeat GNSS spoofing detection mechanisms based on the authentication of ephemeris data, authentication of raw subframes or verification of data consistency with respect to almanac values.

The Galileo I/NAV message stream contains words with non-predictable bits that are utilized for the method proposed in this paper (refer to Section IV for details). The words containing non-predictable bits are present in the data stream every 30 seconds, therefore navigation data prediction cannot be used. In this case, a navigation data relaying strategy for spoofing the I/NAV message stream has to be taken.

2) Navigation data relaying

Navigation data relaying requires the spoofer to relay navigation data bits received by a GNSS receiver to the simulator. In the loosely synchronized spoofing scenario, the spoofer only has access to the binary protocol and raw subframes after they have been received and parity has been verified. Given that it takes 6 seconds to receive a subframe from GPS L1 C/A and that the binary protocol outputs the subframes asynchronously, the expected delay between the actual GPS navigation stream and the simulated stream is 6 seconds + delays due to binary communication protocol + simulator buffering (common simulators require remote modification commands to be executed at least 400ms before the command's reference time). For Galileo E1-B, the expected delay would be approximately 1 second to obtain a page part + communication and buffering delays + 1 second to regenerate the page part.

Galileo pages are protected with three levels of error coding. Each page contains a cyclic redundancy check (CRC) for error detection; a half rate Viterbi forward error correction (FEC) is applied to the message; and finally the resulting frame is block-interleaved with n columns (where data is written) and k rows (where data is read) in order to provide robustness to the FEC decoding algorithm by avoiding packets of errors.

For I/NAV pages, the block interleaver size is 240 symbols with dimensions of n=30; k=8 [11]. Page error coding therefore imposes a delay of 1 second on the spoofer before it is able to access the data bits of the page, given a symbol rate of 250 symbols / second for the E1-B.

As long as the target receiver only tracks spoofed satellites and does not verify the GNSS time with an independent clock, the spoofing attack would most likely be successful.

If the targeted GNSS receiver is able to acquire satellites from the actual constellation as well as the simulation, a number of consistency issues will arise. In order to avoid large errors in the navigation solution, the spoofer would have to assure that a beginning of subframe for each spoofed satellite is within approximately 20ms of those received from actual GNSS satellites. This is because time delays from the satellites are in the range of 67ms (20192 km/c) to 86ms (25785 km/c), where c is the speed of light. If the user is on the surface of the earth, the maximum differential delay time from two different satellites should be within 19 (86–67) ms. Therefore, if there is a differential delay greater than 19 ms, is likely that the receiver is tracking both legitimate and spoofed GNSS signals.

The spoofer could delay the navigation bit stream by a subframe in order to remove the excessive delay observed from the frame synchronization function of the receiver. As the time of week (TOW) of a given subframe increments every 6 seconds, it is assumed that all subframes will be received within the epoch of a given TOW. Most receivers therefore do not check the consistency of the TOW across subframes received on each channel. In order to detect late subframes, the GNSS receiver can simply check that the time of week (TOW) of the subframe received on each channel is equal.

C. Tightly synchronized navigation data relay scenario

Figure 5 illustrates the configuration of a receiver-spoofer and a spoofing scenario for a single channel, where the spoofer attempts to synchronize the simulated signal with that of the real GNSS.

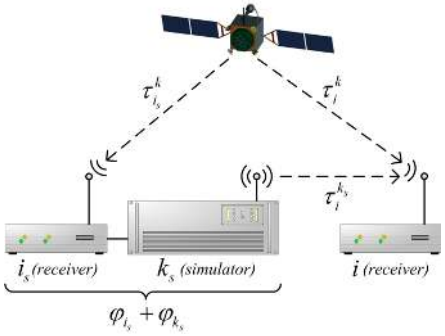

Figure 5. Tightly synchronized spoofing scenario

This type of spoofing attack requires purpose built hardware as well as significant technical / engineering expertise. The cost of such equipment however, has been significantly reduced in the past few years due to the availability of open-

source Software Defined Radio (SDR) software such as GNU radio[1] and low-cost hardware such as the Universal Software Radio Peripheral (USRP) from Ettus Research[2]. This subsection discusses the characteristics of a tightly synchronized navigation data relay attack.

Let i_s denote the spoofer's receiver, k_s denote the spoofer's satellite and φ the processing delay (simulator buffering, etc.) of the spoofer. The receiver time for both non-spoofing (t_i) and spoofing $(\hat{t}_i)$ scenarios can be represented by Equations (1.2) and (1.3) respectively.

$$t_i = t^k + \tau_i^k \quad (1.2)$$

$$\hat{t}_i = t^k + \tau_{i_s}^k + \varphi_{i_s} + \varphi_{k_s} + \tau_i^{k_s} \quad (1.3)$$

The delay due to spoofing is illustrated in Figure 6, where a delay of at least one code period + processing delay and buffering is observed (Galileo E1-B has a code period of 4ms and GPS C/A has a code period of 1ms). The reason for this delay is explained below in terms of the GPS L1 C/A signal.

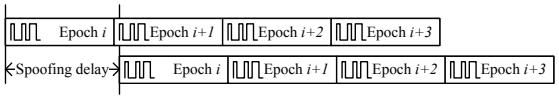

Figure 6. Delay due to spoofing for GPS (1ms) and Galileo E1-B (4ms)

As the authentic signal arrives with a signal power below thermal noise, the individual BPSK modulated chips of the spreading code cannot be easily observed and must therefore be correlated with a locally generated replica code in order to de-spread the signal. This delay (φ_{i_s}) typically accounts for at least 1ms (the period of the GPS C/A code). In a single 1ms frame, correlation with the correct PRN will result in a correlation peak. If the C/N0 is low, the correlation peak may be obscured by noise. GPS receivers can correlate the signal over several frames (1ms code periods) in order to increase sensitivity. Tracking sensitivity can be further increased by the incoherent sum of frames accumulated by a coherent integrator. For the purposes of spoofing, however, it is important to obtain the code phase in the minimum time possible (i.e. 1 frame).

For GPS, access to real-time code phase information can be easily obtained through software in the baseband processor by accessing the in-phase correlator registers that provide the code phase for each channel. Many GPS baseband processors generate an interrupt synchronized to the channel code generation epoch timing.

While Galileo E1-B has a primary code period of 4ms [11], 1ms frames can be correlated with partial segments of the primary code. For the purposes of determining the code phase for spoofing, the code phase could be obtained from the partial correlations. The correlation peak of such partial correlations may be obscured by noise in low C/N0 situations.

For both GPS C/A and Galileo E1-B, a spoofing threshold of 1ms is defined. This threshold is used as a higher bound for the spoofing detection scheme, where it is assumed that achieving synchronization with authentic GNSS signals below these thresholds would require a significantly increased engineering effort and cost.

Processing delay and buffering of the simulation accounts for the remaining component of the spoofing delay (φ_{k_s}). If the spoofer is attempting to synchronize the simulation of GNSS signals such that they are within ½ a chip of the authentic signal (in order to prevent the loss of code lock), an additional code period would be required (i.e. 1ms for GPS C/A and 4ms for Galileo E1-B). Assuming that the spoofer is not concerned with synchronizing the simulation to within ½ a chip of the actual signal, buffering of the simulation could account for as little as a few microseconds (a few code chips).

Similarly to the loosely synchronized navigation data relay scenario, if the target receiver only tracks spoofed satellites, the spoofing attack would most likely be successful. As the time delay of the spoofed signal is negligible (milliseconds) in the tightly synchronized scenario, precise measurement of the time drift would be required.

If the targeted GNSS receiver is able to acquire satellites from the actual constellation as well as the simulation, consistency issues in the navigation solution will arise. Matlab was used to simulate the impact of delays in various channels on the resulting navigation solution. The first subframe index (milliseconds from start of tracking) for each channel in the digitized signal was delayed by up to 10 milliseconds. Table 1 illustrates the results of a series of navigation solutions computed using the least-squares method, where spoofing was simulated with a delay of 1 millisecond (1 GPS C/A code epoch) for all channels and various mixed cases. The results indicate that unless all channels are delayed, significant changes in the clock bias and in particular, the height of the navigation solution can occur when some (not all) of the channels have been delayed.

Navigation Solution	Nominal (6 channels)	Delay of 1 code epoch for all channels	Delay of 1 code epoch for first 3 channels	Delay of 1 code epoch for only 1 channel	Delay of 1 code epoch for all but 1 channel
X	4.4723e6	4.4723e6	4.0163e6	4.1967e6	4.4100e6
Y	6.0141e5	6.0141e5	6.3828e5	4.2402e5	5.4894e5
Z	4.4926e6	4.4926e6	5.0529e6	4.3274e6	4.1851e6
dt_i	5.7039e5	5.7038e5	4.4580e3	3.6496e5	3.1813e5
Latitude	45.0652	45.0653	45.1160	45.9356	43.4819
Longitude	7.6588	7.6589	9.0300	5.7695	7.0955
Height (m)	182.9706	175.1129	**-6.2601e5**	**-3.2409e5**	**-2.6356e5**

Table 1. Simulated spoofing and resulting navigation solutions using least-squares method

In the following section, we propose a method for the detection of both the loosely and tightly synchronized spoofing scenarios described.

IV. Proposed Spoofing Detection Technique

This section discusses the proposed spoofing detection method consisting of a combination of techniques that can provide effective detection of the spoofing scenarios described in Section III, whilst being suitable for implementation in mass-market receivers. Individual techniques such as navigation message authentication and monitoring of the time drift cannot provide protection against the spoofing scenarios described in Section III. However, these methods, combined with the requirement for timing of non-predictable data bits, can

[1] http://gnuradio.org/

[2] http://www.ettus.com/

provide a robust spoofing detection solution. As parts of the navigation message are non-predictable, a spoofer is constrained to relay navigation data bits in real time.

Navigation message authentication, by means of observations by a trusted observer, provides the receiver with a guarantee that the navigation messages have not been tampered with in order to facilitate spoofing. This allows the receiver to trust the ephemerides and clock correction terms contained within the navigation message.

By timing the reception of a subframe containing non-predictable bits, it is possible to determine if the GNSS time has drifted beyond the threshold for spoofing (nominally 1ms), with respect to an independent undisciplined clock. This assumes that the undisciplined clock has been synchronized to GNSS time in a secure way and that the oscillator is sufficiently stable to maintain accuracy between subsequent receptions of subframes containing non-predictable data bits. In order for a given channel to be validated, the non-predicable data bits must be verified by means of navigation message authentication. Figure 7 illustrates the logical processing blocks and data flow of a trusted GNSS receiver implementing the proposed techniques.

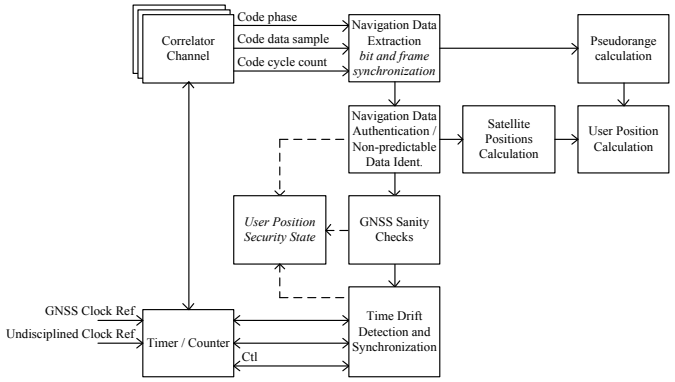

Figure 7. Baseband processing of trusted GNSS receiver

In terms of hardware requirements, the technique requires an additional, undisciplined clock and a baseband processor with at least two timer channels. Three additional functions need to be implemented in the firmware of the baseband processor:

A) navigation data authentication and identification of non-predictable bits; B) GNSS sanity checks; and C) time drift detection and synchronization. These functions are described in the following subsections.

A. Authentication of Navigation Data and Non-predictable Bits

This section discusses navigation message authentication of GPS subframes or Galileo words containing non-predictable bits. It is assumed that a spoofer will not be able to simulate in real-time subframes containing non-predictable bits without a significant time delay (order of milliseconds). In order to find subframes in GPS containing non-predictable bits, an external GPS antenna was installed on the roof of the building housing the Qascom office, allowing raw subframes to be captured for a number of weeks. A GPS receiver with raw measurement capability was used to capture the raw subframes, including those not used by civilian GPS receivers. The following subsections discuss subframes containing non-predictable bits that were found in the GPS naviation message, and Galileo words with non-predictable bits based on currently available documentation.

1) GPS L1 Navigation Data Validation

GPS navigation subframes were analyzed for the presence of non-predictable data bits that could be used to authenticate GPS satellites from simulated ones. Two types of non-predictable subframes were identified: subframes containing the Navigation Message Correction Table (NMCT) and specific "Reserved for System Use" data. The following subsections discuss in more detail the characteristics of the non-predictable data observed in GPS navigation data subframes.

a) Encrypted NMCT data for WAGE

Wide Area GPS Enhancement (WAGE) is a method to increase the horizontal accuracy of the GPS encrypted P(Y) code by adding additional pseudorange correction data to the satellite broadcast navigation message. WAGE data is encrypted, making it available only to the Precise Positioning Service (PPS) or P(Y) code receivers. WAGE data are transmitted in page 13 of subframe 4 of the GPS navigation data (NAV), referred to as the Navigation Message Correction Table (NMCT) [12]. Each NMCT contains an availability indicator (AI) that indicates one of the following availability options:

1. The correction table is unencrypted and is available to both authorized and unauthorized users;
2. The correction table is encrypted and is available only to authorized users (*normal mode*);
3. No correction table available for either authorized or unauthorized users; or
4. Reserved.

The NMCT contains pseudorange error estimations for 30 slots that correspond to the 30 satellites in ascending order excluding the transmitting satellite (Table 2). The estimated range deviation (ERD) fields are encrypted.

W1		W2		Word 3				W4		W5		W6		W7		W8		W9		W10		
TLM	Parity	HOW	t + Parity	DATA ID	SV (PAGE) ID	**AI**	**ERD 1-3**	Parity	**ERD 3-7**	Parity	**ERD 7-11**	Parity	**ERD 11-15**	Parity	**ERD 15-19**	Parity	**ERD 19-23**	Parity	**ERD 23-27**	Parity	**ERD 27-30**	t + Parity	Total
24	6	22	8	2	6	2	24	6	24	6	24	6	24	6	24	6	24	6	24	6	22	8	300

Table 2. GPS NAV Subframe 4 (Page No. 13 NMCT)

During normal operations, the clock and ephemeris data sets (subframes 1, 2 and 3) are transmitted by a satellite for a period of 2 hours with a corresponding curve fit interval of 4 hours. In order to utilize this information for the purpose of validating the authenticity of navigation data, it is imperative that the AI bits indicate the correction table is encrypted and that the NMCT is valid. Only one satellite that did not transmit encrypted NMCT was observed (PRN 32). This satellite's AI field indicated that the NMCT was not available. The contents of the NMCT in this case were predictable and should therefore not be considered for validating the authenticity of navigation data.

The NMCT from a 12-channel GPS constellation simulator was also analyzed. By default the NMCT was disabled on all PRNs. Once enabled, the NMCT was simulated with unencrypted data. In order to ensure that plausible data generated by a spoofer can be detected, valid NMCT data must be authenticated by means of a trusted observer (i.e. navigation message authentication). The validity of the NMCT data can be calculated by performing the following calculations [12]. The age of data offset (AODO) term, which is transmitted in subframe 2, must be less than 27900 seconds in order for the NMCT from the transmitting satellite to be valid. If the term is equal to 27900 seconds, the NMCT is invalid and shall not be used. If the NMCT is valid, the validity time t_{nmct} can be computed as illustrated in Equation (1.4).

$$\begin{aligned} &offset = t_{oe} \bmod 7200 \\ &\text{if } offset = 0,\ t_{nmct} = t_{oe} - AODO \\ &\text{if } offset > 0,\ t_{nmct} = t_{oe} - offset + 7200 - AODO \end{aligned} \tag{1.4}$$

where t_{oe} is the ephemeris epoch time reference.

The value must then be corrected for beginning and end of week crossovers as illustrated in Equation (1.5).

$$\begin{aligned} &\text{if } t^k - t_{nmct} > 302400,\ t_{nmct} = t_{nmct} + 604800 \\ &\text{if } t^k - t_{nmct} < -302400,\ t_{nmct} = t_{nmct} - 604800 \end{aligned} \tag{1.5}$$

where t^k is the time of transmission in GPS time.

The NMCT validity time must be verified with respect to the current ephemeris epoch reference in order to insure that the NMCT data is non-predictable for the current ephemeris epoch. It is possible that t_{nmct} of the current NMCT is older than an ephemeris epoch (nominally 7200 seconds), and therefore is predictable on an ephemeris data crossover (i.e. when Issue of Data Clock (IODC) / Issue of Data Ephemeris (IODE) values change). The trusted observer used for navigation message authentication should provide the time the current NMCT was first observed in addition to providing authentication of the subframes containing the NMCT terms. It is noted in [12] that NMCT information is supported by the block IIR GPS satellites only when operating in the IIA mode of operation.

b) Reserved for system use messages

During the observation of navigation messages for non-predictable data, a number of undocumented messages were observed. In particular, a number of pages from subframe 4 did not appear to be deterministic. According to the Interface Control Document (ICD) [12], subframe 4 is subcommutated 25 times; the 25 versions of these subframes are referred to as pages 1 through 25 of each subframe. With the possible exception of "reserved for system use" pages and explicit repeats, each page contains different specific data in words 3 through 10. (Refer to Table 20-V of the GPS ICD [12])

The undocumented pages 1, 6, 11, 16 and 21 (each with SV page ID 57), identified as "reserved" in the ICD, were monitored for a number of weeks to determine the repetition rate and predictability of data transmitted over this period.

Table 3 illustrates the results of one week of analysis of the "reserved for system use" messages. The data found in these messages was identical for pages 1, 6, 11, 16 and 21.

Day of Week	Week N°	Reserved Field 1 (128-bits)	Reserved Field 2 (16-bits)
Mon	*n*	7E9CF7839C08F1E4EB5F06850FE6E94A	0103
Tue	*n*	F913438213CAFD84A3A15F3A653BBE62	0203
Wed	*n*	343DC30C3DCB4E70326A57C8DAEAF09A	0303
Thur	*n*	80C16989C175F6023321DF32606C7DC6	0403
Fri	*n*	FF9EDB629E8590F577808AFDC517BF33	0503
…	…	…	…
Mon	*n* + 1	A8F59BA2F56EE5B6D055B44F3745F693	0104
Tue	*n* + 1	318E83FC8E3B1E778D8D37671FDEF564	0204

Table 3. Data monitored in pages 1, 6, 11, 16 & 21 of subframe 4

Two distinct fields were observed in the 144 bits of reserved data. Field 1 changes daily and does not appear to have any predictable pattern in the samples collected over a week. Field 2 appears to be a counter that increments every time field 1 changes. The 8 MSBs increment daily, 8 LSBs increment weekly.

Due to the lack of information, further attempts were made to analyze the value using a 12-channel GNSS constellation simulator running in "turbo-mode". The values observed in the navigation message were "AAAAAAAAAAAA…" for field 1 and "AAAA" for field 2 for any TOW and WN. It appears as though these fields are related to restricted or classified functionality, possibly over-the-air keying with a crypto-period of 24 hours, as such data fields are not simulated on GPS simulators for civilian use. We therefore assume that the data in this message cannot be easily predicted and simulated before it is transmitted by genuine GPS satellites.

2) Galileo Safety of Life Service Data Validation

A number of words in the I/NAV message structure contain non-predictable bits that can be used to differentiate authentic Galileo navigation messages from simulated ones. The I/NAV message structure is transmitted on both E5b-I and E1-B signals in order to support a dual frequency service [11], implicitly providing support for the proposed spoofing detection method on both frequencies.

A subframe on the E1-B signal has a duration of 30 seconds, where the chipping rate of the signal is 250 symbols / second. The following subsection details the words in the identified I/NAV pages that are suitable for verification of navigation message authenticity.

a) SoL Integrity Table

In order to support the Galileo Safety of Life (SoL) service on the E1-B signal, integrity tables are broadcast in nominal I/NAV pages every 30 seconds. These tables consist of flags that indicate the integrity status of the signal broadcast by each satellite for a given integrity satellite navigation frame. The flags indicate either: SV (satellite vehicle) not OK for use or SV OK with a given value of SISMA (signal in space monitored accuracy) [13]. A mechanism is additionally provided to support the authentication of the integrity tables. Preliminary information of SoL service indicates that authentication data changes each integrity satellite navigation frame (30 second epoch) and is non-predictable.

B. GNSS Sanity Checks

Before the time delay verification can be performed, the following sanity checks need to be executed to ensure the integrity of the time delay calculations.

- Verification that TOW of subframes across all channels are equal;
- Verification that the maximum differential delay time between pairs of satellites is less than or equal to 19ms (refer to III.B.2); and
- Verification that the height from the navigation solution is within defined thresholds (e.g. 0-4000m, threshold is adjusted with respect to application requirements). A digital barometric altimeter could be used by the receiver to validate the height with respect to a defined threshold, although this would not be useful for self-spoofing, as such devices can easily be tampered with. As described in III.C, delays of at least 1ms observed on some (not all) channels tend to result in significant errors in the clock bias, and in particular the height of the navigation solution. This effect is negligible if all channels are delayed.

The above sanity checks serve to identify if there are both simulated and genuine GNSS satellites present in the navigation solution. If these sanity checks pass, it is likely that either all satellites are spoofed, or all satellites are genuine. Therefore, the receiver can proceed with time delay verification based on the assumption that the clock bias calculation will be reliable. If the checks fail, the presence of spoofing is flagged.

C. Time Drift Detection and Synchronization

As discussed in Section III, spoofing by relaying navigation message bits, it is likely that a delay of at least 1ms will be introduced into the navigation message stream of the spoofed signal. While navigation bits can be predicted, it is assumed for this method that certain GPS subframes / Galileo words cannot be predicted, and therefore must be relayed. Refer to Section IV.A for details of which bits of GPS NAV and Galileo I/NAV cannot be predicted.

Equation (1.6) illustrates how the time delay incurred due to spoofing can be observed using an independent clock, not disciplined by GNSS.

$$t_{uc} - t_i = \tau_{i_s}^k + \varphi_{i_s} + \varphi_{k_s} + \varepsilon_{uc} \quad (1.6)$$

where, t_{uc} is the GNSS time obtained from an undisciplined clock; t_i is the receiver GNSS time; $\tau_{i_s}^k$ is the travel time between the spoofer and the targeted receiver (i.e. if the distance is 1 meter, the travel time will be approximately 3.3356×10^{-6}ms); φ_{i_s} is the delay incurred by receiving the data bit (i.e. de-spreading of signal); φ_{k_s} the delay incurred due to processing and buffering for retransmission by the simulator; and ε_{uc} is the local undisciplined clock error, which needs to be less than $\varphi_{i_s} + \varphi_{k_s}$ if the signal is to be authenticated. The relationship between the GNSS time of transmission and the GNSS time at the receiver is illustrated in Equation (1.7).

$$t_i = t^k + \left(P_i^{k_s} / c\right) \quad (1.7)$$

where t_i is the receiver's GNSS time and P_i^k is the psuedorange observation whose relationship between the geometric range ρ_i^k, receiver clock bias dt_i, satellite clock offset (code phase and equipment delay terms) dt^k, tropospheric delay T_i^k, ionospheric delay I_i^k and pseudorange observation error e_i^k is illustrated in Equation (1.8).

$$P_i^k = \rho_i^k + c(dt_i - dt^k) + T_i^k + I_i^k + e_i^k \quad (1.8)$$

The time drift between the local undisciplined clock and the GNSS time for the purpose of authentication can be obtained from the absolute GNSS sample clock cycle count (start of code index) corresponding to the beginning of an authenticated subframe (or Galileo word) containing non-predictable data bits.

The receiver clock bias, dt_i, and position are determined through the linearization of a nonlinear equation solving four unknowns (the receiver ECEF position X_i, Y_i, Z_i and dt_i) for a set of at least four pseudoranges using a method such as the least-squares method (refer to Section II). It is imperative that the equation is solved using authenticated TOW (Time of Week) (for satellite positions which require time of transmission corrected by travel time), satellite clock offset values, ephemerides, tropospheric and ionospheric delay correction terms.

Once dt_i is known, the time offset between the GNSS time at which the beginning of a subframe containing non-predictable data bits is received and the GNSS time derived from the local undisciplined clock can be calculated. The GNSS time at the beginning of subframe refers to the time at which the rising-edge of the first chip of the first code of the first data bit of the subframe is received.

Figure 8 illustrates the logical blocks of two general purpose timer channels used to calculate the time offset between the GNSS receiver's reference clock (clock used to sample GNSS signal) and the local undisciplined clock. Both timers are configured with a comparator that resets its corresponding counter once the frame epoch has been reached. The epoch interrupt of the GNSS reference clock counter is used by the acquisition engine, correlators and the 1PPS (1 pulse per second) generator. In order to keep track of the current epoch, an epoch count register is incremented when an interrupt is raised by the comparator. Multi-channel timers are commonly found in microcontrollers and a number of GNSS baseband processors.

The time offset between the GNSS reference clock *rclk* and the undisciplined clock *uclk* in clock cycles can be calculated for a given time *t* using Equation (1.9).

$$\Delta t(rclk, uclk) = \left(EpochCount_{rclk}(t)\cdot n\right) + CycleCount_{rclk}(t) - \left(EpochCount_{uclk}(t)\cdot n\right) + CycleCount_{uclk}(t) \quad (1.9)$$

where $EpochCount_{rclk}(t)$ is the epoch count of the GNSS reference clock at time *t*; $CycleCount_{rclk}(t)$ is the cycle count

of the GNSS reference clock at time t; $EpochCount_{uclk}(t)$ is the epoch count of the undisciplined clock at time t; $CycleCount_{uclk}(t)$ is the cycle count of the undisciplined clock at time t; and n is the number of clock cycles per epoch (frame epoch is typically a multiple of 1023 for GPS L1 C/A or 4096 for Galileo E1-B).

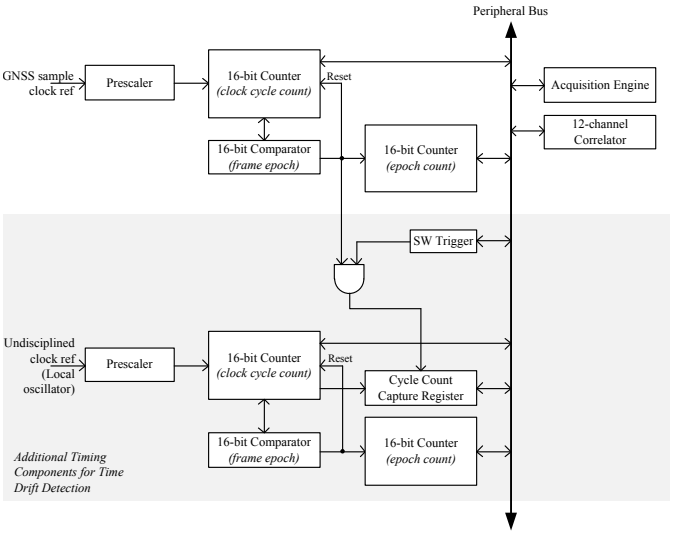

Figure 8. Time drift detection timer configuration

In order to achieve a constant time t across timer channels, a cycle count capture register for the undisciplined clock counter is configured to latch when the software trigger has been activated and the GNSS reference clock counter reaches frame epoch.

Once the offset between the two clocks is known, the absolute cycle count of the undisciplined clock for the beginning of subframe ($AbsCycleCount_{uclk}(bsf)$) can be calculated using Equation (1.10).

$$AbsCycleCount_{uclk}(bsf) = \left(EpochCount_{rclk}(bsf) \cdot n\right) + CycleCount_{rclk}(bsf) + \Delta t(rclk, uclk) \quad (1.10)$$

where $EpochCount_{rclk}(bsf)$ is the epoch count of the GNSS reference clock at the beginning of subframe (bsf); $CycleCount_{rclk}(bsf)$ is the cycle count of the GNSS sampling reference clock at the beginning of subframe; $\Delta t(rclk, uclk)$ is the offset between the two clocks; and n is the number of cycles per epoch. The beginning of subframe cycle and epoch count values are obtained from the bit and frame synchronization function, which in turn obtains these values from the prompt correlator.

The GNSS time based on the undisciplined clock for the beginning of subframe, $t_{uclk}(bsf)$, is therefore calculated as follows:

$$t_{uclk}(bsf) = AbsCycleCcount_{uclk}(bsf) + GnssTimeOffset_{uclk} \quad (1.11)$$

where $GnssTimeOffset_{uclk}$ is the GNSS system time offset from the local undisciplined clock in cycles, obtained during a secure time transfer or resynchronization process.

The presence of spoofing due to the delay of the navigation message stream for a given satellite k can be determined using Equation (1.12).

$$t_{rclk}(bsf) = t^k(bsf) + \left(P_i^{k_s}/c\right)$$
$$SignalAuthentic_t(k) = \left(t_{uclk} - t_{rclk}\right) < \left(\varphi_{i_s} + \varphi_{k_s}\right) \quad (1.12)$$

Where $t_{rclk}(bsf)$ is the receiver time at the beginning of subframe, $t^k(bsf)$ is the time of transmission from satellite k at the beginning of subframe obtained from the Time of Week (TOW) field of the subframe and the clock correction terms provided in the navigation message; and the threshold $\varphi_{i_s} + \varphi_{k_s}$ = 1ms for GPS C/A and Galileo E1-B (refer to Section III.C for discussion on data relay spoofing scenario).

A channel is deemed authentic when $SignalAuthentic_t(k)$ is true for the reception of a subframe with non-predictable data bits that have been authenticated by a trusted observed. Once a channel transitions to the "authenticated" state, the channel state remains as "authenticated" until there is a loss of frame / bit synchronization or the local clock time drift exceeds the window of acceptance.

On a successful authentication event, the clock offset $GnssTimeOffset_{uclk}$ can be updated using Equation (1.13).

$$GnssTimeOffset_{uclk} = \left(t_{rclk}(bsf) \cdot m\right) - AbsCycleCount_{uclk}(bsf) \quad (1.13)$$

where $t_{rclk}(bsf)$ is the receiver time at the beginning of subframe, m is the milliseconds to clock cycles conversion factor and $AbsCycleCount_{uclk}(bsf)$ is the absolute cycle count of the undisciplined clock at the beginning of subframe. The navigation solution can be flagged as trusted when it is calculated from the observations of SVs (satellite vehicles) that have been verified for time drift. Both bit and frame synchronization for each channel must be monitored. If bit or frame synchronization is lost for a given channel with a previous state of "authenticated", the channel security state must be reset to an "authenticity unknown" state.

The degree to which detection of a spoofing delay is possible depends on the accuracy of the local clock. Table 4 illustrates the approximate times before different types of oscillators drift beyond 1ms.

Crystal Oscillator	Approx frequency deviation (PPM)	Approx time till 1ms drift	Approx Cost
XO	10 – 100	100 – 40 secs	< $1
TCXO	0.5 – 2	33.3 – 8.3 mins	$1 - $50
MCXO	0.01 – 0.1	27.7 – 2.7 hours	< $1000
OCXO	0.01 – 0.5	27.7 – 0.55 hours	$200 - $2000

Table 4. Comparison of crystal oscillators [14] (Note that costs have been updated)

Table 5 compares the maximum frequency deviation of an oscillator in order to not drift by more than $\varphi_{i_s} + \varphi_{k_s}$ for series of time periods. The suitability of the various oscillators for time drift detection is detailed in Table 4 for each time period.

GPS L1 subframes containing non-predictable data bits can be observed every 2 hours during normal operations (refer to the GPS ICD [12] for details of the other operational modes),

whereas Galileo words containing non-predictable data bits can be observed every 30 seconds.

Time period (*not suitable for GPS)	Maximum frequency deviation in order to not drift by more than $\varphi_{i_s} + \varphi_{k_s}$ (1ms)	Crystal Oscillator			
		XO	TCXO	MCXO	OCXO
1 minute*	16.6666 PPM	Y	Y	Y	Y
30 minutes*	0.5556 PPM		Y	Y	Y
2 hours	0.1388 PPM			Y	Y
4 hours	0.0694 PPM			Y	Y
1 day	0.0115 PPM			Y	Y
1 week	0.0016 PPM				

Table 5. Suitability of crystal oscillators for undisciplined clock for GPS L1 and Galileo E1-B authentication

The application of this method to GPS would be satisfactory for fixed (non-mobile) applications including secure time distribution (e.g. NTP servers), but not very practical for mobile applications, as failure to demodulate data bits for a sufficient number of channels in a data cross-over boundary would result in loss of time synchronization and therefore require secure time transfer before the position and time could be trusted. Secure time transfer involves setting the GNSS time offset value for the undisciplined clock (GTO_{uc}) using secure methods that cannot be spoofed. Such methods include authenticated network time synchronization protocols that are able to synchronize the undisciplined clock to within $\varphi_{i_s} + \varphi_{k_s}$ of GNSS time (1ms for GPS L1 and Galileo E1-B). Note that GTO_{uc} is in GNSS time and therefore is not corrected for UTC.

Without an initial secure time transfer, the receiver will not be able to perform time drift verification and subsequently determine if the navigation solution can be trusted. Once a secure time transfer has occurred, the receiver is able to periodically synchronize the undisciplined clock when authenticated subframes with non-predictable data bits are acquired. Self-synchronization only works while the clock drift between the GNSS reference clock and the undisciplined clock is less than the spoofing delay $\varphi_{i_s} + \varphi_{k_s}$.

Self-synchronization involves the periodic synchronization of the undisciplined clock to the GNSS receiver clock when a GPS L1 subframe or Galileo E1-B I/NAV word with non-predictable data bits is observed, and successfully authenticated. The synchronization can only take place if the drift between the GNSS reference clock and the undisciplined clock is less than the spoofing delay $\varphi_{i_s} + \varphi_{k_s}$.

In order to ensure that the undisciplined clock does not drift beyond the acceptance window (less than $\varphi_{i_s} + \varphi_{k_s}$), the receiver can periodically wake itself up via a timer at intervals when subframes with non-predictable data bits are expected in order to perform self-synchronization. These subframes must also be authenticated by a trusted observer.

For example, for a Galileo OS receiver, the wakeup timer could be set for every 30 minutes if a TXCO (Temperature Compensated Crystal Oscillator) is used for the undisciplined clock, or even daily if a MCXO (Microprocessor Controlled Crystal Oscillator) or OCXO (Oven-Controlled Crystal Oscillator) is used. Self-synchronization is less effective for GPS, due to the long periods between the reception of subframes with non-predictable data bits. In the case of a GPS, the receiver should either wake up every 2 hours, or remain powered on.

V. Authentication Services and Receiver Operating States

This section discusses the requirement for authentication services required to support the navigation data authentication function and operating states of a trusted GNSS receiver implementing the methods discussed in this paper.

Figure 9 illustrates a GNSS broadcast authentication service comprised of one or more trusted observers, networks over which broadcast authentication messages are transmitted, and receivers that are able to receive the broadcast messages.

The suitability of network technologies for the authentication of subframes / words depends on the characteristics of the GNSS application. For example, mobile applications could make use of network technologies such as GPRS (General Packet Radio Service of GSM) or DVB (Digital Video Broadcasting), whereas static applications could additionally make use of fixed network technologies such as DSL (Digital Subscriber Line). Existing networks used for the broadcast of GNSS augmentation data such as EDAS (EGNOS data access service) are potentially ideal candidates for the transmission of GNSS authentication data. Details of protocols to facilitate the exchange of authentication data between the GNSS receiver and the trusted observer, and key distribution and public key infrastructure issues are outside the scope of this paper.

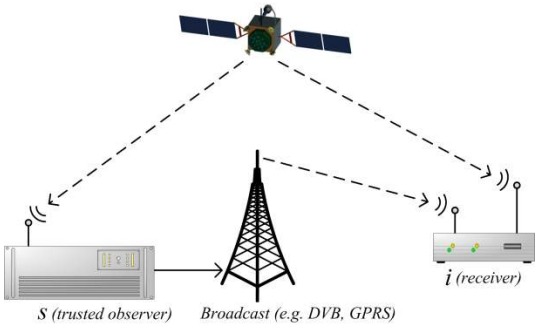

Figure 9. GNSS authentication broadcast service

A GNSS receiver implementing the methods discussed in the preceding sections would be able to provide an alarm indicating the presence of spoofing and possibly a trusted navigation solution. The trusted GNSS receiver would start in an “unknown” state. In a cold start scenario, receiver’s undisciplined clock offset (GTO_{uc}) has not been set and the receiver must therefore be synchronized with a trusted time source. In this case, the receiver transitions to the “time transfer required” state.

Once secure time transfer has been performed, the receiver transitions to the "unknown” state until it is able to verify the authenticity of the GNSS signals or detect spoofing, resulting in transition to the “authenticated” or “spoofing detected” state. The presence of at least 4 authentication channels is sufficient to transition to the “authenticated” state. If bit synchronization is lost for a given channel, the security state for that channel is reset to “unknown”.

If the clock drift between the undisciplined clock and the GNSS clock exceeds the spoofing delay threshold, and the projected time drift is less than the threshold, spoofing is detected. In the case the projected time drift is equal to or greater than the threshold, indicating that self-synchronization was unsuccessful (e.g. due to low C/N0), the receiver transitions to the "time transfer required" state. The projected time drift is an estimation of the time drift based on the characteristics of the crystal oscillator, and is used to minimize false positives. The high-level states of receiver GNSS authenticity are illustrated in Figure 10. The receiver could operate in one of two modes, depending on the application:

- *an alarm-only mode*: such that the navigation solution is assumed to be derived from authentic GNSS satellite signals until a spoofing alarm is raised;
- *a secure navigation mode*: such that a navigation solution is only provided when sufficient channels have been authenticated, and the navigation solution comprising of the 4 authentication channels is within a threshold the mixed solution (the threshold being compatible with application performance requirements).

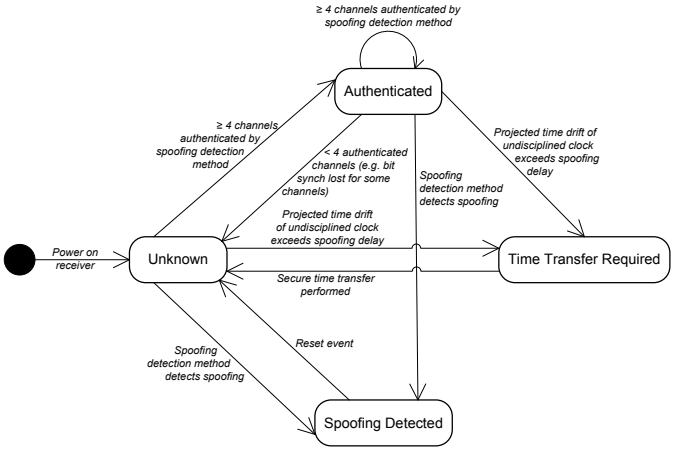

Figure 10. Receiver GNSS authenticity state diagram

VI. Conclusion

In conclusion, this paper has discussed several spoofing scenarios and has presented a low-cost spoofing detection method suitable for numerous mass-market applications. The method exploits non-predictable characteristics of GPS L1 C/A and the Galileo E1-B Safety of Life Service in combination with navigation message authentication and time delay verification to provide protection against the spoofing scenarios discussed.

VII. Acknowledgments

Oscar Pozzobon from Qascom S.r.l. and Markus Kuhn from the University of Cambridge are gratefully acknowledged for their valuable comments. Part of the work was performed within the Trusted Innovative GNSS receivER (TIGER) project[3], co-funded by the European GNSS Agency (GSA), grant agreement n° 228443.

[3] http://www.tiger-project.eu

VIII. References

[1] Green Bay Professional Packet Radio (GBPPR), "GPS Delay Spoofing Experiments," *The Monthly Journal of the American Hacker*, no. 53, September 2008.

[2] John A. Volpe National Transportation Systems Center, "Vulnerability Assessment of the Transportation Infrastructure Relying on Global Positioning System," 2001.

[3] Logan Scott, "Anti-Spoofing & Authenticated Signal Architectures for Civil Navigation Systems," *In Proceedings of ION GPS/GNSS 2003*, September 2003.

[4] Chris Wullems, Oscar Pozzobon, and Kurt Kubik, "Signal authentication and integrity schemes for next generation global navigation satellite systems," *In Proceedings of the European Navigation Conference (ENC-GNSS 2005)*, July 2005.

[5] Markus G. Kuhn, "An Asymmetric Security Mechanism for Navigation Signals," *In Proceedings of the 6th Information Hiding Workshop*, May 2004.

[6] Sherman Lo, David De Lorenzo, Per Enge, Denis Akos, and Paul Bradley, "Signal Authentication: A Secure Civil GNSS for Today," *Inside GNSS*, pp. 30-40, September / October 2009.

[7] Jon S. Warner and Roger G. Johnston, "GPS Spoofing Countermeasures," Los Alamos National Laboratory, LAUR-03-6163, 2003.

[8] Hengqing Wen, Peter Yih-Ru Huang, John Dyer, Andy Archinal, and John Fagan, "Countermeasures for GPS signal spoofing," *Proceedings of the 18th International Technical Meeting of the Satellite Division of the Institute of Navication (ION GNSS 2005)*, 2005.

[9] Todd E. Humphreys, Brent M. Levinda, Mark L. Psiaki, W. O'Hanlon, and Paul M. Kintner, "Assessing the Spoofing Threat: Development of a Portable GPS Civilian Spoofer," *Proceedings of ION GNSS, The Institute of Navigation*, 2008.

[10] ARINC Research Corporation, "GPS Interface Control Document," ICD-GPS-200C, April 12, 2000.

[11] European Space Agency / European GNSS Supervisory Authority, "Galileo Open Service Signal In Space Interface Control Document," OS SIS ICD, Draft 1, February 2008.

[12] ARINC Engineering Services, LLC, "Navstar GPS Space Segment/Navigation User Interfaces," IS-GPS-200 Revision D, December 7, 2004.

[13] C Pecchioni et al., "Galileo Test User Receiver Development for Safety of Life Applications: Integrity Processing Overview," in *ENC-GNSS*, Naples, Italy, 2009.

[14] John R. Vig, "Introduction to Quartz Frequency Standards," Army Research Laboratory Electronics and Power Sources Directorate, Fort Monmouth, NJ 07703-5601, U.S.A., SLCET-TR-92-1 (Rev. 1), 1992.