# Protection and Fundamental Vulnerability of GNSS

Panagiotis Papadimitratos
EPFL
Lausanne, Switzerland
panos.papadimitratos@epfl.ch

Aleksandar Jovanovic
EPFL
Lausanne, Switzerland
aleksandar.jovanovic@epfl.ch

***Abstract*—An increasing number of mobile applications and services require that devices are aware of their location. Global Navigation Satellite Systems (GNSS) are the predominant enabling technology. But location information provided by commercial GNSS is *not* secure, unlike what is the usual assumption. There are only few exceptions in the literature that present GNSS vulnerabilities. In this paper, we contribute the first detailed quantitative analysis of attacks against GNSS-based localization. We show how *replay attacks* against GNSS can have a significant impact: even against cryptographically secured GNSS instantiations, an adversary can manipulate the location and time calculated by victim GNSS receivers. We explain in detail how such attacks can be mounted, measure their impact, and discuss the effectiveness of possible countermeasures.**

## I. Introduction

A plethora of mobile devices require knowledge of own position or position of other devices in the system. Examples include sensors reporting environmental measurements, cellular telephones or portable digital assistants (PDAs) and other portable computers, embedded mobile platforms as those in Vehicular Communication (VC) systems, and merchandize (container) and fleet (truck) management systems. In these cases, mobile devices determine their own position, and other system entities determine the position of the mobile devices. These two aspects are the two main categories of positioning (localization) problems. In this paper, we are concerned with the former, the determination of own position; more specifically, we consider *Global Navigation Satellite Systems* (GNSS) as the enabling localization technology, since GNSS receivers are used in numerous and diverse applications.

The Global Positioning System (GPS), its Russian counterpart (GLONAS), and the upcoming European GALILEO system transmit signals bearing reference information from a constellation of satellites. Any receiving device, $V$, with the appropriate equipment can decode the signals and utilize the GNSS information to determine its own location, $loc_V$. However, publicly known wireless transmission and reception methods for GNSS signals and message content create a vulnerability. They open the door to system abuse by an adversary that interferes with and injects fictitious GNSS transmissions, and this way manipulates $loc_V$ and eventually $V$'s operation. For example, consider an attack against a fleet management system, with trucks traveling across a continent, e.g., Europe, equipped with GPS and reporting periodically their location via a cellular telephone data link to a central server. If the adversary wishes to physically attack a specific truck with a GNSS receiver $V$, it can transmit forged GPS signals in the vicinity of $V$, cause a false $loc_V$ to be calculated and then reported, and mislead the fleet management system about the actual location of its truck. In a different setting, an adversary could affect the navigation systems running nowadays on many (private and public) vehicles, disrupt the routes drivers are provided with, and eventually degrade the efficiency and safety of the transportation system.

Such attacks exploiting the vulnerability of commercial GNSS positioning can have serious consequences. With the increasing popularity of those systems, it is very important to analyze and thwart GNSS abuse. In this paper, we investigate exactly this problem, notably, the impact of such attacks and possible ways to mitigate them. A few works only considered the GNSS vulnerability, discussed in [4], [7], and countermeasures are proposed in [3], [6], [8]. Our contribution here is two-fold: We provide a quantitative analysis of attacks through detailed simulations, and we identify the significance of *replay attacks*, a class of relatively simple to implement attacks that can be mounted even against future cryptographically protected systems. We emphasize that to this date, commercial GNSS systems do not provide authentication services; this is a feature of the upcoming GALILEO system.

In the rest of the paper, we first provide an overview of basic GNSS characteristics and outline attack types in Sec. II. We describe security mechanisms considered in the literature in Sec. III. Then, in Sec. IV, we explain the specifics of replay attacks and in particular why they can be successful in spite of cryptographic protection. Before we conclude, we discuss in Sec. VI the effectiveness of several mechanisms to protect GNSS receivers.

## II. Vulnerability of GNSS Signals

### A. GNSS overview

Each receiver is able to receive simultaneously a set of *navigation messages*, one $NAV_i$ message from each satellite $S_i$ in the visible *satellite constellation*. Each satellite is assigned an a priori publicly known *unique spreading code* $c_i$. The $NAV_i$ enable each receiver $V$ to determine its own *position*, $loc_V = (X_V, Y_V, Z_V)$, in a Cartesian system, as well as a *time correction offset*, $t_V$, to add to its local clock value in order to maintain the current global time. At least four satellites should be visible so that $V$ can compute $loc_V$ and $t_V$, with the two quantities together termed the ***PVT*** or *navigation solution*. The computation at the receiver relies on *pseudo-range* values it estimates, one per visible satellite $S_i$: the pseudo-range $\rho_i$

978-1-4244-1948-7/08/$25.00 © 2008 IEEE 

is the $S_i$-$V$ distance estimate, based on the satellite signal propagation delay. This is calculated as the difference of $V$'s local clock at reception time, minus the time at which $NAV_i$ was transmitted, provided by $S_i$ as a field in $NAV_i$. For each pseudo-range $\rho_i$, one equation is formed:

$$\rho_i = |s_i - loc_V| + c \cdot t_V \quad (1)$$

The position, $s_i$, of satellite $S_i$ is obtained from the $NAV_i$ message, and $c$ is the speed of light. With a system of at least four equations, $V$ obtains the PVT solution [1].

### B. Attacking GNSS

The essence of the attacks against commercial GNSS lies in that the satellites' spreading codes are publicly known. This allows an adversary to construct a transmitter of signals identical to those sent by a satellite. The objective of the adversary would then be to *forge* NAV messages, transmit them over an area with one or more receivers, and this way manipulate their PVT solutions. Nonetheless, with GNSS signals being widely (essentially, globally) available, the adversary should first force its victim receivers to loose their "lock" on legitimate GNSS signals and then "lock" on the adversary's forged signals.

To mount such an attack, the adversary should act essentially in two stages. First, it should *jam* the GNSS signals, to force receivers to loose contact with the satellites, and then transmit its forged messages. The latter is termed a *spoofing* or a *meaconing* attack, depending on whether the adversary synthesizes its transmissions or "re-uses" (parts of) legitimate GNSS transmissions.

The adversary could mount the second and essential stage of its attack even without forcing receivers to loose their "lock" to GNSS signals. This would be possible when there are gaps in GNSS coverage, that is, areas where $V$ cannot lock on more than three satellites. This may occur often in urban environments and in general due to obstacles that cause loss of GNSS signals. We do not dwell on this case, as loss of satellite signals is not under the control of the attacker.

**Jamming** The attacker transmits with sufficiently high power in the GNSS frequency band. This deliberate interference forces receivers to "unlock," i.e., loose contact with the otherwise visible satellite signals. A *jammer* is a simple, low-cost yet very effective device. For example, with 1 Watt of transmission power, reception of GNSS signals is prevented approximately within a radius of 35 km from the jammer.

**Spoofing** signals can be generated by satellite *simulators*, equipment which is available today. The received power of the spoofing signal should exceed that of the legitimate signal; this being essentially a form of jamming. The receiver then operates with the forged signal as input and computes the location induced by the *spoofer*. For example, it could "invert" the navigation solution it wishes to impose, estimating the $s_i$ positions of the satellite constellation, and then set the corresponding NAV message values. Beyond falsifying the *time of the week number* (ToE) (to influence the receiver's clock) or the *almanac* data, the adversary can perform more subtle manipulations of NAV message parameters such as the *mean anomaly at reference time (Mo)*, which describes the angular offset between the satellite position at reference time and perigee, or the *eccentricity e* and the rate of *right ascension* ($\Omega$) [9]. Alternatively, spoofed signals can be generated based on previously received GNSS signals: the adversary records NAV messages and re-transmits them or it synthesizes new messages with their parts. This attack, termed **meaconing**, is in the class of **replay attacks** that we explain in further detail in Sec. IV.

## III. Securing GNSS Signals

Authentication and integrity protection of NAV messages have emerged as defense mechanisms. Authentication would ensure that NAV messages generated only by GNSS entities are accepted at and used by receivers to determine the PVT solution. Integrity would ensure that modification or utilization of parts of NAV messages towards spoofing or meaconing is not possible either. Next, we discuss different approaches for cryptographic protection.

Symmetric key cryptography, with one secret key shared by the GNSS and each receiver, is impractical: NAV messages would need to be authenticated for each of the many millions of receivers individually. On the other hand, a single, system-wide symmetric key, shared with all receivers would be very efficient but also very weak. The entire system security would be in jeopardy, as it suffices for the adversary to compromise the system key (e.g., physically read out from *one* receiver) and then launch spoofing attacks at will. Making GNSS receivers tamper-resistant currently appears very costly for commercial devices.

Asymmetric or public key cryptography appears as a viable choice given the problem constraints. The GNSS obtains pairs of private and public keys, $k_i$ and $K_i$ respectively, one pair per satellite, with each $K_i$ bound to $S_i$ via a certificate. Each receiver obtains all certified $K_i$ for all GNSS satellites. Each satellite digitally signs with $k_i$ its NAV messages. This service, termed *Navigation Message Authentication* (NMA) [5], will be available in GALILEO.

To further enhance protection against meaconing, a different public-key NMA approach was proposed in [3]. To thwart replay attacks, each $S_i$ chooses a secret spreading code for each NAV message. But it discloses this to receivers, along with a hidden timing marker, in a delayed and authenticated manner. If nodes can maintain accurate clocks by means other than the GNSS methods, they can safely detect messages that are replayed between the time of their creation and their spreading code disclosure. A similar idea using secret spreading codes (SSCs) is presented in [8].

## IV. Revisiting the GNSS Vulnerability: Replay Attacks

With the upcoming NMA and future GNSS security, it is still possible for an adversary to manipulate the receiver's PVT solution. NMA can thwart forgery but it cannot prevent *replay attacks*, a a general class of attacks. The adversary can receive legitimate GNSS signals (and thus NAV messages),

record them, and transmit them at a later point in time and at a different point in space. This is possible because, essentially, cryptography ensures the authenticity and integrity of messages but cannot ensure signal authenticity: a message can be re-transmitted by any radio other than the one of the message originator.

We illustrate a replay attack in Fig. 1. In general, the adversary can start recording the GNSS frequency band after the beginning of the navigation message is detected. This is done thanks to a preamble in the form of an 10001011 sequence of bits. The adversary needs to detect at least the first bit of the preamble; this can be done after a period of $\tau = 20$ms, the transmission delay for one 1 bit with the GNSS bit-rate of 50 bps.[1] After that, the adversary can start replaying recorded signals, with any additional $t_{replay} \geq 0$ delay it chooses. The victim receiver(s) will start receiving the replayed NAV messages after some adversary-victim specific signal propagation delay (usually negligible) and one more bit transmission time.

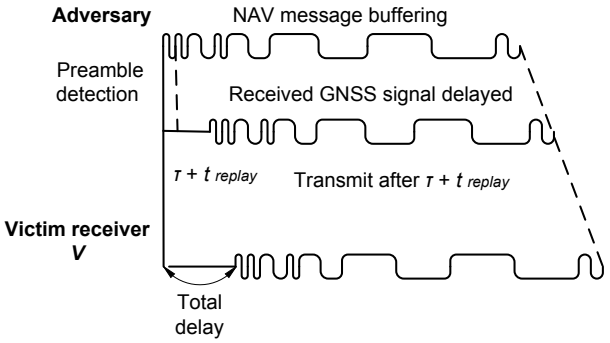


Fig. 1. Illustration of a replay attack: $\tau$ is the minimum processing time to detect a NAV message, and $t_{replay}$ is additional delay imposed by the attacker.

$t_{replay}$ is an essential characteristic of the replay attack. It allows the adversary to control the "shift" of the PVT solution it induces. Looking at the signals from each $S_i$ separately, each millisecond of $t_{replay}$ translates approximately to 300m of erroneous (induced by the adversary) $S_i$-$V$ distance offset: the receiver of the replayed signal will have an erroneous $\rho_i$ estimate, 300m longer than the actual one, as radio signals propagate with the speed of light, $c$. This is exactly why the PVT solution of $V$ will be affected. The adversary can have the choice of which $S_i$ signals to replay and the choice of a $t_{replay}$ value for each one. If it replays all NAV messages "blindly" (the entire band, as explained below), it induces the same $t_{replay}$ to all $NAV_i$. The received signal $I$ *component* after a replay could be represented as:

$$S_{L1} = \alpha C_A(t - t_{replay})D(t)sin(ft + \phi) \qquad (2)$$

In Eq.(2), $\alpha$ represents the amplitude of the signal, $f$ the transmitted frequency, $\phi$ is the received phase, $C_A$ is the $C/A$ code and $D(t)$ is the navigation message. Given the cryptographic protection, $D$ cannot be modified by the adversary. Nonetheless, even though this is not necessary for replay attacks, the adversary has control over $\alpha$ and thus, to a significant extent, the received power at $V$. It also controls $f$ and this way can affect the phase and thus the received signal Doppler offset. We do not consider those actions in further detail here. We note, however, that this relates more generally with the sophistication of the adversary's equipment. For example, the more transceivers it has and the more closely matching signals to the original GNSS they produce, the more effective it can be in targeting a specific victim receiver. Adversaries with different sophistication levels will be considered in our future investigations.

Another main characteristic of replay attacks is the method of GNSS reception, which enables replaying. This can be done at the message or symbol level or it can be done by recording the GNSS frequency band and replaying it without de-spreading the GNSS signals. The latter, more involved and thus costly, would enable the attacker to mount an attack against the delayed-disclosure secret spreading code approach, as pointed out in [3]. But this additional feature, which essentially implies a stronger adversary instantiation, would be necessary for low $t_{replay}$ values. In contrast, this is not necessary for long $t_{replay}$ values, after the de-spreading code for the hidden marker is disclosed. A high $t_{replay}$ allows the adversary to circumvent the delayed disclosure approach, for example, after a prolonged period of jamming or benign loss of satellite signals, or after a "cold start," that is, when the receiver boots after a long period of no operation and thus no memory of its clock and location.

## V. Evaluation of Attack Feasibility and Impact

### A. Simulation Setup and Rationale

In order to evaluate the impact of attacks, we perform a detailed simulation of the system, implementing the basic software functionality of the GPS receiver, and simulating the satellite trajectories in Matlab. We calibrate the simulation environment with raw data from a JPS receiver in RINEX format [11]. We implement jamming of the victim receivers followed by spoofing and replay attacks. We consider only the portion of the network area under the influence of the attacker, and thus the area at which spoofed or replayed GNSS signals are the ones mobile receivers lock on to. We consider a three-dimensional area for receivers, and various mobility models. For the results shown here, the positions of receivers are updated every second, for a total simulation time of 300 seconds. The receiver velocity $\vec{v}$ changes at each time step $t$, with a component randomly drawn from an interval added to $v_{\vec{X}}$, $v_{\vec{Y}}$, and $v_{\vec{Z}}$.[2] We model benign errors, due to multi-path, ionosphere delay, and other random errors in signal propagation, which cause pseudo-range estimation errors and

[1] This is the GPS bit rate. For GALILEO, with rate of 250 bps, the delay would be 4ms.

[2] The random component added at each time step, for the results shown here, is in $[-5, 5]$ km/h for each component. Choosing the initial velocity and the limits for each component, we can control the level of mobility. For example, for high mobility, $v_{\vec{X}}$, $v_{\vec{Y}}$ in $[55, 185]$ and $v_{\vec{Z}}$ in $[15, 30]$.

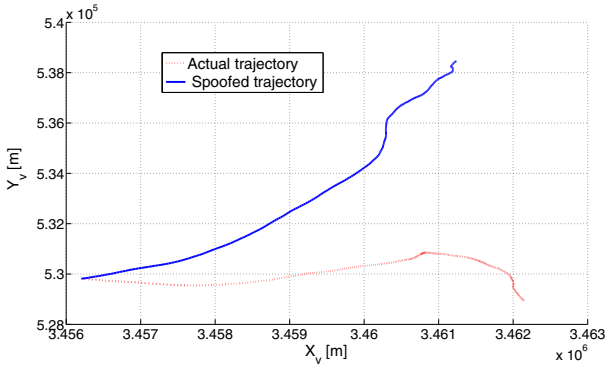


Fig. 2. Replay attack induced location offset for the $loc_V$ of a victim receiver $V$: example actual and spoofed trajectories projected on the X-Y plane.

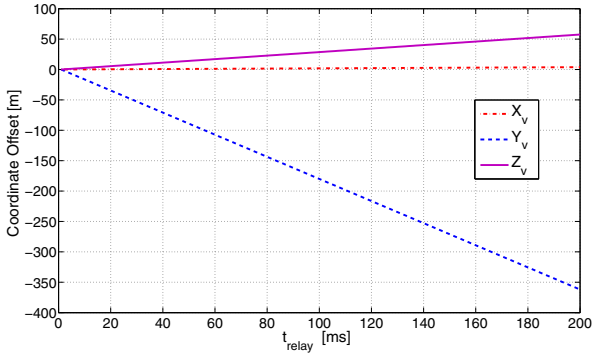


Fig. 3. Replay attack induced location offset for each of the $loc_V$ coordinates $(X_V, Y_V, Z_V)$ of a victim receiver $V$.

thus errors in the navigation solution, via a Weibull distribution with shape factor $b = 2$; a randomly drawn factor is added to each coordinate of $loc_V$ independently [10].

We measure the impact of each attack in terms of the *location offset* the attack causes to the navigation solution, with respect to the actual location of the victim receiver. Since spoofing can be thwarted with cryptographic protection (NMA), and due to space limitations, we do not present here results on their impact. Rather, we focus on the impact of replay attacks. Our objective is to investigate and show the feasibility of *fine-grained replay attacks*. The reason is that in practice very large location offsets could be detected, as discussed in Sec. VI.

We implement the relaying attack as described in Sec. IV, controlling the $t_{replay}$ imposed by the adversary. The remaining components of delay, including $\tau = 20$ms, between the original NAV transmission and reception at the victim are not under the control of the adversary. It is possible that practical equipment constraints could restrict $t_{replay} \geq t_{replay}^{min} > 0$. We allow adversary to add $t_{replay} \in [1, 200]$ms. The adversary can either use the same $t_{replay}$ for all the NAV signals it replays (a case that also corresponds to replaying the spread signals), or it can utilize different $t_{replay}$ values for each $NAV_i$. We vary $t_{replay}$ in steps of 1ms. We also let the adversary choose $t_{replay}$ randomly for each $NAV_i$. For the results shown here, signals from all visible satellites are replayed (and delayed). We experimented with adversaries that replay a subset of visible signals, with the difference explained in Section V-B.

### B. Results

Fig. 2 illustrates the actual trajectory of a mobile receiver versus the spoofed trajectory, which is induced by a replay attack with $t_{replay} = 20$ms; for easier inspection, we project the trajectories on the $X - Y$ plane. We see the discrepancy increasing, after 300 seconds of simulation, ending up with a location offset of 12 km. It is interesting to see that a subtle attack, in the sense that initially the location offset as well as the time offset $t_V$ are low, results gradually in a significant "displacement" of the victim receiver.

Next we investigate the impact of the replay attack as a function of $t_{replay}$, looking at the location offset, for each of the coordinates $(X_V, Y_V, Z_V)$ of $loc_V$. Intuitively, the larger the delay, the larger the offset. This is verified by Fig. 3, which shows a significant offset increase with $t_{replay}$. We see that $X_V$, $Y_V$, $Z_V$ have different error sensitivity; for example, $Y_V$ and $Z_V$ change at the rate of few meters to tenths of meters per second, while $X_V$ changes are much lower. We emphasize that the error (offset) sensitivity per coordinate can vary in different ways, beyond what is shown in Fig. 3, as there is a geometric dependence, to specific ephemeris data and the satellites and receiver positions. We observed overall that the location offset increases with $t_{replay}$; we let the detailed investigation of per coordinate sensitivity as future work.

We finally show results for the adversary that introduces a random $t_{replay}$ per satellite NAV message, randomly chosen here from the $[1, 20]$ms interval. Fig. 4 shows the resultant locations, given an actual position, as a "cloud" of points, and Fig. 5 shows the effect (offset) for each Cartesian coordinate $X_V$, $Y_V$, $Z_V$ over (simulation) time. We observe a sharp difference with the homogeneous replay attack, which caused location offsets in order of meters (for one NAV message). In contrast, the randomized replay attack, even with low $t_{replay}$ values, can severely distort the perceived satellite geometry, and thus create location offsets in the order of hundreds of kilometers (e.g., 500 km). We note here that we experimented with replay attacks that manipulated signals from subsets of satellites, but we do not present those here due to space limitations. Thanks to the least-squares PVT solution, replaying for example two or three signals causes lower location offsets.

## VI. Towards Thwarting the GNSS Vulnerability

As replay attacks allow the adversary to manipulate $loc_V$ for any receiver $V$ in the area its signals overwrite GNSS signals, defense mechanisms are necessary to detect the onset of an attack. In other words, after $V$ enters the area under adversarial control, it should be able to detect whether the $loc_V$ it calculates is the result of the attack. We consider the following basic approach: when the receiver has an indication of an attack, for example, GNSS unavailability or, possibly, jamming detection, it controls the plausibility of the PVT solution once it "locks on" to GNSS signals again. We propose

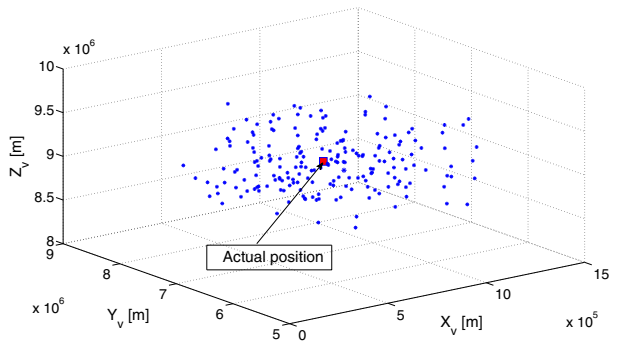


Fig. 4. Replay attack with random $t_{replay}$ values for different satellite signals: attack-induced positions with respect to a given actual receiver position.

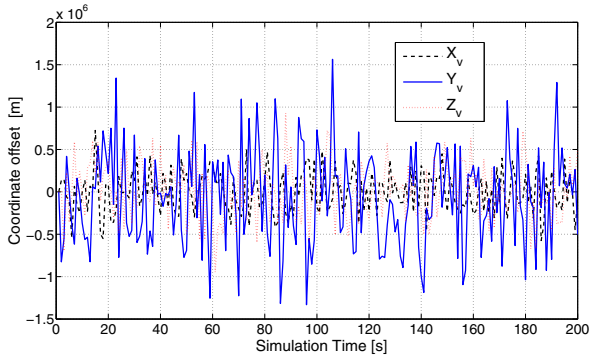


Fig. 5. Replay attack with random $t_{replay}$ values for different satellite signals: attack-induced offsets for each of the $loc_V$ coordinates $(X_V, Y_V, Z_V)$ over part of a simulation trace.

doing this primarily by controlling if the PVT "fits in" with previous location and time values, known to the receiver when there were no indications of an attack.

We propose simple tests to "filter out" replayed messages. $V$ can control if the propagation delay for the $\rho_i$ equations falls in the range of approximately 67 to 87ms, which correspond to the satellite orbits from $20,000$ to $26,000$km. Of course, the adversary could choose $t_{replay}$ such that the resultant total propagation delay estimate remains in the $67-87$ms range. Moreover, the receiver can loosely detect replayed signals by comparing the approximate information in the almanac with the visible satellites: it can detect an attack if there is a mismatch. Again, an adversary of relatively low sophistication could avoid replaying signals of satellites that contradict the almanac.

Regarding memory-based tests, the receiver applies essentially an approach that resembles Receiver Autonomous Inertial Mechanisms (RAIMs). Using past locations, before the suspected attack, it predicts its location. Taking into consideration statistical prediction errors, which increase with the period of disconnection, it compares the suspected $loc_V$ with the predicted one. It detects an attack if the PVT $loc_V$ falls outside the error margins of the predicted value. Similarly, the receiver clock can remain relatively stable even without the PVT-based synchronization. A replay attack causing a high $t_V$ value, outside the maximum receiver clock drift margins, can be detected this way.

We emphasize that these techniques are not effective if the receiver has no memory of time or location, e.g., when it boots after a long period of being non-operational (cold start). Nonetheless, they can be effective for a significant fraction of attacks. We will investigate those further, as well techniques that can mitigate attacks perpetrated during cold start in future work.

## VII. Conclusion

Existing GNSS receivers are vulnerable to a range of attacks that manipulate their computed location. We identify a fundamental GNSS vulnerability to replay attacks, even if GNSS were cryptographically protected. Our quantitative analysis, based on a detailed simulation framework, shows that fine-grained replaying attacks can be performed, so that gradual manipulation of the victim location can remain undetected. But, cumulatively, those small manipulations can lead to a substantial displacement of the victim over a period of time. Without any compromise, physical or not, of the GNSS receiver or other equipment related to the location-aware application, the adversary can attack the system; for example, a cargo can be stolen while in a location away from its believed one. It is important to note that replay attacks can exploit the GNSS functionality without compromise of the GNSS receiver or other node on-board equipment or node-to-node communication. We discuss replay attacks against different future NMA-secured GNSS, with attackers of differing sophistication.

As part of on-going and future work, we intent to further refine and generalize our simulation framework, present additional details on the impact of attacks, considering more closely the cost of attacks of differing sophistication levels through proof-of-concept implementations, and develop further countermeasures and evaluate their effectiveness.

## References

[1] E.D. Kaplan and C.J. Hegarty, *Understanding GPS - Principles and Applications*, Artech House, 1996
[2] N. Bertelsen and K. Borre, *A software defined GPS and Galileo receiver*, Birkhauser, 2007
[3] M. Kuhn, *An asymetric Security Mechanism for Navigation Signals*, 6th Information Hiding Workshop, Toronto, Canada, 2004
[4] J.A. Volpe, *Vulnearability Assesment of the transportation infrastructure relying on GPS*, NTSC NAVCEN draft report, 2001
[5] G.W. Hein and F. Kneissl, *Authenticating GNSS Proofs against Spoofs*, InsideGNSS, September/October 2007
[6] H. Wen, P.Y. Huang, J. Dyer, A. Archinal, and J. Fagan, *Countermeasures for GPS signal spoofing*, Unpublished manuscript
[7] A. Pinker and C. Smith, *Vulnearability of GPS Signal to Jamming*, GPS Solutions, Vol.3, No.2, 1999
[8] L. Scott *Anti-Spoofing and Authenticated Signal Architectures for Civil Navigation Signals*, ION-GNSS, Portand, Oregon, 2003
[9] NAVSTAR GPS Joint Program Office, *NAVSTAR Global Positioning System - Interface Specification IS-GPS 200 Space Segment/Navigation User Interfaces*, SMC/GP, CA, USA, 2004
[10] *http://users.erols.com/dlwilson/gpsacc.htm*
[11] *http://www.bernese.unibe.ch/download.html*