Efficient authentication mechanisms for navigation systems – a radio-navigation case study

Georg T. Becker[1], Sherman Lo[2], David De Lorenzo[2], Di Qiu[2], Christof Paar[1], Per Enge[2]
[1]*University of Bochum,* [2]*Stanford University*

BIOGRAPHY

Georg T. Becker is a PhD student at the University of Bochum in electrical and computer engineering. He received his B.S in applied computer science and his M.S. in IT-Security from the University of Bochum.

Dr. Sherman Lo is a senior research engineer at the Stanford University Global Positioning System (GPS) Laboratory. He is the Associate Investigator for the Stanford University efforts on the Department of Transportation's technical evaluation of Loran.

Dr. David De Lorenzo is a research associate at the Stanford University Global Positioning System (GPS) Laboratory. He received his M.S. in Mechanical Engineering from University of California, Davis and his Ph.D. in Aeronautics and Astronautics from Stanford University.

Di Qiu is a Ph.D. candidate in Aeronautics and Astronautics at Stanford University. She received her B.S in Aerospace Engineering from University of California, Los Angeles and her M.S. from Stanford University. Her current research interests are location-based security services, signal authentication and fuzzy extractors.

Per Enge is a Professor of Aeronautics and Astronautics at Stanford University, where he is the Kleiner-Perkins, Mayfield, Sequoia Capital Professor in the School of Engineering. He directs the Stanford GPS Research Laboratory.

Christof Paar is a Professor in Electrical and Computer Engineering at the University of Bochum where he heads the chair for Embedded Security. His research interest includes efficient software and hardware implementations of cryptographic algorithms, sidechannel attacks, cryptanalytical hardware, and security in real-world systems.

ABSTRACT

In this paper we introduce an efficient authentication mechanism especially designed for navigation systems that is based upon the Timed Efficient Stream Loss-Tolerant Authentication (TESLA) algorithm. We analyze the different attack scenarios on navigation systems and show that it is only necessary to authenticate the source and time of the signals to enable a secure position determination. Traditional message authentication is only needed to prevent counterfeit correction message attacks. With this knowledge and a detailed security analysis of the needed key size, we developed adjusted TESLA, an authentication mechanisms that can authenticate the source and time-messages using only 80 bits. One of the reasons why we can use such a small authentication message is due to the insertion of a timestamp into the generation of the one-way chains. This significantly increases the security of adjusted TESLA compared to the original TESLA and enables us to use a smaller key size.
Adjusted TESLA has a about a 75% smaller size than traditional digital signatures that have signature sizes of at least 320 bits. To prevent counterfeit correction message attacks additional 32 or 40 bits are needed for the transmission of a MAC. But this is still an improvement of at least 62.5% compared to digital signatures or the first proof-of-concept implementation of TESLA in eLORAN. This enables us to significantly improve the security of navigation systems by using only a very small data rate. We propose the use of adjusted TESLA in eLORAN. With this security improvement and LORAN's strength against over-the-air attacks, eLORAN will not only be a backup for current GNSS systems, but will be a real alternative for current civil GNSS systems in application that require the highest possible security level against attacks.

1.0 INTRODUCTION

The use and application of navigation systems, particularly satellite navigation, has grown tremendously in recent years. Today, the number of Global Navigation Satellite System (GNSS) users number hundreds of millions and it is used in even the most safety critical applications. With the growth have come improvements to its availability, accuracy and integrity. However, there has not been a commensurate improvement in the security of navigation systems. With the progress in technology, equipment suitable for attacking GNSS is becoming increasingly cheap. GNSS jamming of wide areas can be done easily and cheaply. GNSS spoofing is possible with small handheld devices and there is anecdotal evidence that several spoofing incidents have occurred, for example [14]. These threats will continue to increase. This is because GNSS use is growing, particularly for monetarily valuable commercial activities such as road toll or electronic tag systems. GNSS spoofing can aid pirates by allowing them to trick their intended target away from their convoy or their protectors. These are just some of the many examples of lucrative attack scenarios.

Satellite navigation systems are especially vulnerable to attacks, due to their weak signals. Terrestrial systems, such as LORAN, have the big advantage of using low-frequency and high power signals. Spoofing or jamming such a signal is much more difficult, as more power and bigger antennas are needed. Nevertheless, this only increases the complexity of an attack but does not prevent them completely. Currently, LORAN is getting updated to enhanced LORAN (eLORAN) as a backup for GNSS. This is a great opportunity to include security mechanisms into eLORAN, so that it can be used in applications in which attack-resistant positioning is needed.

The outline of this work is as followed. At first the different attacks against navigation signals and the general countermeasures against these attacks are described. In the third section an efficient authentication mechanism called adjusted TESLA is introduced. The security of adjusted TESLA is discussed in the next section. The implementation of adjusted TESLA and the resulting authentication times are described in section 5. The thesis is completed by a conclusion in section 6.

2.0 NAVIGATION SECURITY AND LORAN

Jamming: A jamming attack is the most trivial, but also the most common attack on positioning systems. A jamming attack is a denial-of-service attack (DOS) against positioning systems. The attacker tries to interfere with the signals/messages transmitted by the navigation system, so that the receiver cannot determine a correct position. Jamming attacks are a great threat to navigation systems, especially to satellite based navigation systems such as GPS or GALILEO. This is due to the fact that the received signals from the satellites are very weak. Jamming a terrestrial positioning system like eLORAN is much harder compared to jamming a satellite based positioning system. This is due to the fact that LORAN uses high-power, low-frequency signals. To jam LORAN, the attacker needs to overcome the strong LORAN signal. However, to efficiently transmit a low-frequency signal, physically tall or long antennas are needed [6].

Signal-synthesis attack: In a signal-synthesis attack, an attacker generates and sends out false navigation signals to make the receiver believe it is at a different position. If the structure of the navigation message is known to the public, an attacker can easily create valid navigation messages. For example, an attacker can simply attach a power amplifier and an antenna to a commercial civilian GPS signal simulator, which are used for testing GPS receivers, to broadcast the false navigation signals.

Relaying attack (wormhole attack): The basic idea of a relaying attack is to relay the navigation signals received at the wanted spoofing position p' to the receiver at the actual position p. This will make the receiver believe to be at the false position p', although the true position of the receiver is p. There are different ways to execute a relaying attack. If the attacker has physical access to the receiver, an attacker can dismount the antenna and connect the receiver to an antenna located at the false position p'. If the distance between the wanted spoofing location and the actual location is very big, an attacker will very likely not directly attach the antenna at the false position p' to the receiver but will use another channel to transmit the received signals at position p' to the receiver at position p. This kind of attack is also often called wormhole-attack. While designed for other, more friendly, purposes, GPS repeaters can be used to perform a wormhole attack.

Selective-delay attack: In most RF-based positioning systems, the position is calculated by either time of arrival (TOA) of the navigation signals, or the time difference of arrival (TDOA) of the navigation signals from different transmitters. In time of arrival, the receiver uses the time it took the navigation signal to reach the receiver to compute the distance between the transmitter and the receiver. By computing the distance between three different transmitters, the receiver can compute its three-dimensional (3-D) position. To determine the travel time of the signals, the receiver needs to be synchronized with the transmitter stations. If the receiver's clock is not synchronized with the transmitter station, the receiver needs a fourth signal to determine the 3-D position and exact time. Hence, 4 signals are needed to determine the three-dimensional position, as in most cases the receiver will not be synchronized with the transmitter station. TDOA works similar to TOA. The difference is that the

receiver does not use the arrival time of one signal to compute the position but the difference in arrival times between two signals from two different transmitters. In a selective-delay attack, an attacker takes advantage of the fact that the arrival time of the navigation signals are used to determine the position. An attacker delays each navigation signal in a way that the receiver calculates a false position. In this way the attacker can spoof the receiver for the entire coverage area of the signals.

Counterfeit correction message attack: Positioning systems can be effected by several different error sources. To minimize the effects of these error sources, reference stations are used to collect data and to generate correction messages that will help the receiver to calculate a more accurate position. Famous examples for such augmentation systems for GPS are the differential GPS (DGPS), the Wide Area Augmentation System (WAAS), or the Local Area Augmentation System (LAAS). eLORAN will also support correction messages to be able to achieve the required accuracy and availability for applications such as harbor entrance and approach navigation. The correction messages are either transmitted over a different channel, like it is done in DGPS, WAAS or LAAS, or are part of the data message transmitted by the positioning system, like it is planned in eLORAN. The limits of the correction in these messages are around 200-600 meters. In practice, the limits are rarely approached and the actual transmitted correction only changes the calculated position for a few meters or tens of meters.
In a counterfeit correction message attack, the attacker forges these correction messages. A receiver using these forged correction messages will compute a false position. Besides directly forging the correction message, the attacker can also try to attack the reference stations. If the attacker is able to successfully spoof a reference station, this station will generate false correction messages, causing the receivers to compute false positions.

Shifting the tracking point: Another way of attacking a positioning system is described in [6]. The important information in positioning systems is the exact arrival time of a signal. The arrival time of a signal is defined by a tracking point. In the case of LORAN (and eLORAN), this tracking point is defined as the sixth zero crossing of a LORAN pulse. The attacker can try to overlay a signal on the original LORAN pulse, so that the receiver will falsely detect a wrong tracking point. In this way, the attacker does not need to overcome the signal power so that much less power is needed for the attack. Overlaying a signal over the original signal might result in a false envelope shape of the corresponding signal and therefore might be detectable.

Tamper the receiver: If the attacker has access to the receiver, he can try to tamper the receiver. There are various different ways how the attacker can try to tamper the receiver. This strongly depends on the attack scenario. For example, an attacker can change the firmware of the receiver, so that false positions are calculated or manipulate the display of a receiver.

2.1 Countermeasures against attacks on navigation systems

There are several countermeasures against the different attacks. These countermeasures differ in complexity and in the amount of provided security.

Signal and data observation: Signals sent out by the attacker, instead of the valid transmitters, very likely differ in several properties. By monitoring and comparing these properties, the receiver might be able to detect the forged signals. Signal analysis helps to defend against signal-synthesis, selective-delay and relaying attacks. However, more sophisticated attacks will more likely cause these countermeasures to fail. The big advantage of signal analysis is that it can be implemented on the receiver side. Hence, these countermeasures do not require changing the positioning system. The effort needed to implement the countermeasures differs a lot. For example, an IMU cross check needs expensive hardware and complex calculation and is only reasonable in high-security applications. On the other hand, checking for jumps in time and space can easily be done without additional hardware.
The most promising signal and data observation technique is the angle of arrival check. Using arrayed antennas, the receiver can determine the angle of arrival of the incoming signals. If all signals are transmitted from the same transmitter, the receiver will reject these signals. Therefore, the attacker needs several spoofing devices located in such a way that the receiver accepts the angle-of-arrival of these signals. Besides the logistical difficulty of setting up several spoofing devices, the attacker also needs to synchronize the transmission of each spoofing device. Hence, angel-of-arrival discrimination is one of the most promising countermeasures against spoofing attacks.

Message Authentication: The goal of message authentication is to prevent an attacker from generating his own navigation messages. This will make signal-synthesis attacks and counterfeit-correction message attacks impossible. In most cases, message authentication is a requirement for further countermeasures such as hidden markers. Message authentication can be achieved in three different ways, encryption with an integrity check, message authentication codes (MAC) and digital signatures. In this paper a very efficient message authentication mechanism called adjusted TESLA is introduced.

Hide the signal: A very powerful countermeasure against nearly all attacks is to hide the navigation signal in the noise level. Only the users with the correct key can reveal and use the navigation signals. The best example for this technique is the military GPS P(Y) code. The P code is encrypted by multiplying the 10.23 MHz P code with the secret 500kHz W code to form the military Y code. As a result, the signals peak power-spectral density is reduced by about 53 dB and ends up roughly 28 dB below the thermal noise density seen by a typical receiver [3]. Hence, in both, the time and frequency domain, the Y signal disappears in the noise. This encrypts the signal similar to a stream cipher. Without the correct spreading sequence Y it is not possible to reveal the navigation signal.
This makes signal-synthesis attacks impossible, as an attacker will not be able to generate these signals without the secret code. Hidden signals can also prevent selective-delay attacks. To successfully launch a selective-delay attack, each navigation signal of the different transmitters needs to be delayed for a different amount of time. To delay each signal for a different amount of time, the signals from each transmitter station need to be separated from each other. However, if the signals arrive at the same time, this will only be possible if the secret code is known or if the signals can be raised above the noise level.
The big disadvantage of hidden signals is the need of a symmetric key. If the spreading sequence is known to an attacker, the security of this system would be entirely broken. Therefore, this countermeasure can only be used in trusted user groups such as the military. Otherwise, the secret key might be published. These receivers also need to be tamper proof, as otherwise the secret code might be revealed by tampering a receiver.

Hidden markers: The idea of hidden markers was introduced by Scott in [2] and Kuhn in [3]. Hidden markers are used to prevent selective-delay attacks. The main idea is to hide signals, called hidden markers, in the noise level. These hidden markers can only be recovered using a secret key. This key will be released after some delay *d*. The user digitizes and buffers the entire antenna input so that the hidden markers and the exact arrival time of the hidden markers are stored. After the delay *d*, the key will be published and the receiver can reveal the hidden markers and the exact arrival time in the recorded noise using this key. If the arrival times of the hidden markers match with the navigation signals, the signals are valid. To prevent attackers from creating their own hidden markers with a different key, the key needs to be authenticated, e.g. by using a digital signature. This makes hidden markers an asymmetric security mechanism, as the user only needs to know the public key of the used signature scheme. The user does not need to know any secret information to be able to authenticate the hidden markers.

3.0 AUTHENTICATION MECHANISMS

One way to prevent signal synthesis attacks is to authenticate the data messages of eLORAN. The Timed Efficient Stream Loss-tolerant Authentication algorithm (TESLA) can be used to efficiently authenticate the data messages and was proposed for the use in eLoran in [4].

3.1 TESLA

TESLA uses Message-Authentication-Codes (MAC) to authenticate the data messages. A MAC is the symmetric counterpart to a digital signature. The input to a MAC is a secret key and the data message. The output is a tag of a fixed length. Without the knowledge of the key it is not possible to generate or validate this tag.
The basic idea of TESLA is to authenticate the messages with MACs and to reveal the corresponding keys after some delay. An entity will not be able to authenticate the message until it receives the corresponding key. After the key is published, the receiver can use this key to verify the MAC and the corresponding message. Obviously, after the key has been published, an attacker could use this key to generate a valid MAC. However, a receiver only accepts MACs before the corresponding key has been released. To be able to reject old MACs, the transmitter and the receiver need to have at least loosely synchronized clocks.

The key for each MAC is authenticated using a self authenticating one-way chain. In the following, the generation of the one-way chain and how this one-way chain can be used to authenticate keys is discussed.

Key-chain generation: To build a one-way chain, a one-way function H is needed. In a one-way function, the computation of an output y=H(x) should be easy, while the inverse computation of a value x for a given y should be computationally infeasible. In practice, hash functions are often used, so that one-way chains are often called hash chains.

To generate a one-way chain of length l, at first a random number r is chosen. The first node K_l is computed using the random number and the one-way function H(), with K_l=H(r). Then the rest of the nodes are generated by computing K_i=H(K_{i+1}) for $i=l-1$ to $i=1$.
In TESLA, the private keys are the nodes of the one-way chain $K_l,...,K_2$. The public key is the last computed key K_1 and the time intervals during which each key is valid. The transmitter reveals the keys in the reverse order of the computation of the key. So the key that is released first is K_2, the next key is K_3 and the last one is K_l. When a receiver receives a key K_i at time T_i, the receiver can validate the key by comparing the output of $H(K_i)^{i-1}=K_1$ with the public key. If these values are the same and the time interval for key K_i matches the current time, the key

is valid. The one-way chain generation and publication is visualized in figure 1.

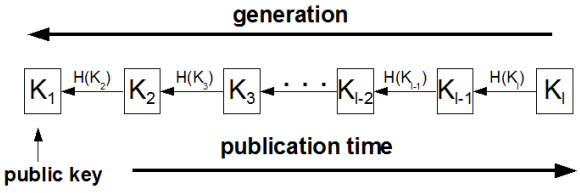

Figure 1: The generation and publication of a TESLA one-way key chain.

Signature generation: To sign a message m_i during the time interval T_i, the transmitter generates a MAC $MAC_{K_{i+1}}(m_i)$ with the key K_{i+1}, which will be revealed at some time after the current time interval T_i. During the time interval T_i, the transmitter sends out the message m_i, the MAC $MAC_{K_{i+1}}(m_i)$ and the key K_i which is supposed to be released during this time interval. The receiver cannot authenticate the message m_i yet, because he does not know the key K_{i+1}. During the time interval T_{i+1}, the transmitter sends the current message m_{i+1}, the MAC $MAC_{K_{i+2}}(m_{i+1})$ and the key K_{i+1}. The receiver now validates the key K_{i+1} by computing $K_1=H^i(K_{i+1})$ and comparing this value with the public key. After the receiver has verified the key, the receiver can verify the authentication of the message m_i by computing $MAC_{K_{i+1}}(m_i)$. Hence, the receiver can only authenticate messages after some delay d. This delay d is defined by the time until the needed key is revealed. This delay is a security parameter, as it defines the allowed maximal offset between the clock of the receiver and the transmitter. Figure 2 illustrates the verifying process.

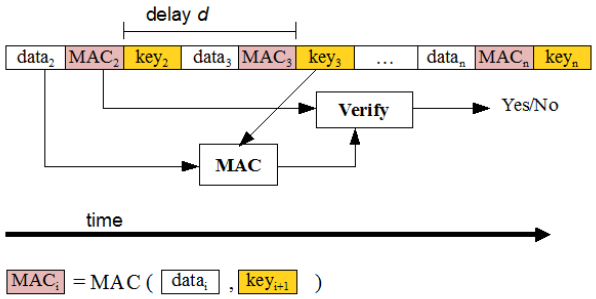

Figure 2: The basic structure of TESLA.

3.2 Adjusted TESLA

Traditionally, TESLA is used to authenticate data messages. However, the content of the data messages is not the most important part in positioning systems. In some cases, this data only provides correction data and time information that might not be essential for the position determination. Message authentication is needed to defend against counterfeit correction message attacks, but it is not necessary to have message authentication to prevent signal-synthesis attacks. To protect against signal-synthesis attacks, it only needs to be ensured that the signals cannot be generated by an attacker.

In most navigation systems, only two pieces of information are needed to determine a position, the source and transmission time of the signal. If the receiver can be sure that the received signals comes from the assumed transmitter stations and knows the exact transmission time, the receiver can securely compute the position without the need of additional information. To prevent a signal-synthesis attack, the signal needs to be somehow unpredictable for an attacker so that an attacker cannot generate valid signals. Transmitting a valid key from the one-way chain can be used as source authentication. Based on the design principles of one-way chains, no unauthorized entity should be able to generate valid keys for a one-way chain. Hence, only the valid transmitter is able to reveal a key. The only option an attacker has to transmit a valid key is to replay an old key. As one requirement of TESLA, the clocks of the receiver and the transmitter need to be loosely synchronized. If the attacker uses a key older than the delay d, the receiver will detect that the key is not valid. However, an attacker can delay current keys with a delay which is small enough such that the receiver does not get suspicious. Such an attack is not a signal-synthesis attack but a delay attack. This type of attack is always possible if only message authentication is used. Hence, this attack would also be possible if a MAC or a digital signature is used. Because of that, the MAC does not improve the security against signal-synthesis attacks, selective-delay or relaying attacks. It only improves the security against counterfeit correction message attacks.

It is possible to authenticate the time message and the station ID without the use of a MAC. The station ID is a fix value and each transmission time of a signal can be predicted. This makes it possible to embed the station ID and the transmission time into the key generation of the one-way chain. In the original TESLA, a time interval is given for each key during which the key is valid. It is possible to tighten this time schedule, by defining the exact transmission time of each key. One way to do this, is to exactly define the time interval between two keys and publish the transmission time of the first key of the one-way chain. For example, assume that the first key is revealed at 00:00:00 and every second one key is revealed. If you receive the key k_{73}, you know that this key was sent 73 seconds - 1 second =72 seconds after the first key, so the transmission time was 00:01:12. Note that not every key needs to be revealed by the transmitter. If some of the previous keys have not been published, the

current key can still be validated. Therefore, it is possible to generate a key for every possible transmission time in case that the exact transmission time of each key is not known, e.g. because the data messages vary in size.
To make sure that the time schedule is known to the receiver, it is possible to embed a timestamp into the one-way chain generation. To generate a one-way chain, at first a random number r is chosen, like it is done in the original TESLA. The nodes are generated by adding the timestamp of the transmission time of each node to the input of the one-way function. The first node K_l is computed by using the random number r and the timestamp $time_l$ as an input to the one-way function: $K_l=H(r,time_l)$. The rest of the nodes are generated by computing $K_{i-1}=H(K_i,time_i)$ for $i=l$ to $i=2$, where $time_i$ denotes the timestamp of the transmission of K_i.(e.g. $time_i$ is the eLORAN time message) The generation and publication of a one-way chain with a timestamp is illustrated in figure 3.

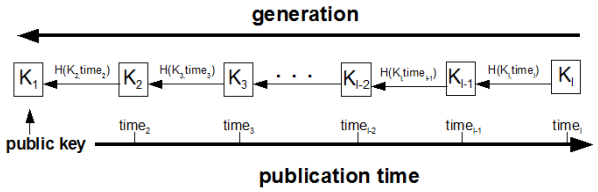

Figure 3: The generation and publication of an adjusted TESLA one-way key chain with an embedded timestamp.

To validate a key K_i, the receiver needs to know the transmission time T_i of K_i. As the transmission time is used to compute the position, the receiver already knows the transmission time. The receiver computes the timestamp $time_i$ for the time T_i. With this timestamp, the receiver computes the previous key $K_{i-1}=H(K_i,time_i)$. For $j=i-1$ to $j=2$ the receiver computes $K_{j-1}=H(K_j,time_j)$. If K_1 matches the public key, K_i is valid and was sent by the transmitter at the time T_i. As we will see in section 4.1, adding a timestamp significantly increases the security of the one-way chain. Besides the idea of only using a one-way chain to authenticate the signals, this embedded timestamp is the other main contribution in this paper.
To make sure that the message comes from the assumed station, each station should have their own one-way chain and the station ID should be added to the generation of the one-way chain.(e.g. as part of the timestamp) This ensures that the receiver knows which station sent out the key. In our implementation we use the eLORAN time message as the timestamp. We will generate a key for every second, regardless whether or not a key will be published in this second. This ensures that for every possible transmission time a key will be available. (The transmission of one eLORAN data packet takes more than one second so that there can never be two messages within one second)

The big disadvantage of using only the one-way chain with a timestamp and no MAC is the missing message authentication. How important message authentication is depends strongly on the positioning system, the application and the type of messages. Two things need to be considered, while deciding whether or not message authentication is needed.

1. How difficult is a counterfeit correction message attack in case only a one-way chain and no MAC is used compared to other attacks such as selective-delay or relaying attacks?
2. How dangerous is a counterfeit correction message attack compared to these attacks? Are there attack scenarios that are only possible with counterfeit correction message attacks?

In most cases, a counterfeit correction message attack is at least as difficult as a selective-delay attack. In a counterfeit correction message attack the attacker sends out forged correction messages. These forged correction messages need to be synchronized with data messages containing the one-way chain keys. To send out forged correction messages that are synchronized with the authentication messages at the receiver is very challenging and in eLORAN as difficult as a counterfeit correction message attack.

The second question, whether or not a counterfeit correction message attack is more dangerous than a selective-delay attack, is very important as well. In most cases, the same goals that can be achieved with a counterfeit correction message attack can also be achieved with a selective-delay attack. Moreover, selective-delay attacks can spoof a position within the entire coverage area of the received signals, while counterfeit correction message attacks can only introduce a much smaller error to the actual position.

However, if the LORAN data messages contain additional information that is not directly used by eLORAN to determine the position, attacks that are not possible with a selective-delay attack might become feasible. For example, if the LORAN data channel is used to broadcast correction messages for GPS, like it is proposed in [13], an attacker could compromise eLORAN and GPS at the same time by attacking the LORAN data channel. Therefore, it would be more important to secure the data messages, as an insecure LORAN data channel might compromise the security of other systems in this case.

Sometimes it also depends on the situation whether or not message authentication is needed. For example, if correction messages only correct your position for less than two meters, saving bandwidth by not sending MACs might be reasonable. But if correction messages make a difference of more than 50 meters, securing the correction

messages might be beneficial. Another example where message authentication might be needed is when a transmitter station has a malfunction and sends out false navigation data. In this case, the other stations send out warning messages which should be secured, as using the malfunctioned transmitter can cause great position offsets. However, in absent of these abnormalities, it might not be necessary to secure the correction messages. The data channel might not only be used to transmit data for the positioning system, but could also be used to transmit warning messages, for example hurricane or tsunami warnings. For these rare events, message authentication might be necessary as these warnings might have great impacts. Furthermore, forging these messages enable attack scenarios which are not possible with a selective-delay or relaying attack. To be able to switch between message authentication and no message authentication would be the ideal solution. Reference [1] describes how this can be realized.

4.0 SECURITY OF ADJUSTED TESLA

Navigation systems have only a very small data rate. In eLORAN, the data rate depends on the used GRI. In the worst case, it takes about 2.4 seconds to transmit a 120 bit LORAN data packet. However, of these 120 bits 75 bits are parity bits, 4 are header bits and only 41 are payload data bits. This results in a worst case data rate of only about 17 bits/second. Therefore, it is important to choose an authentication mechanism with as little overhead as possible. A trade-off between security and key size is needed. The key size should be chosen so that the security of the system is ensured, while it as small as possible to have short authentication times. A security analysis of adjusted TESLA shows that, due to the insertion of the timestamp during the generation of the one-way chain, adjusted TESLA is much more secure than traditional TESLA. Therefore a small key size of only 80 bits can be used for adjusted TESLA. For comparison, traditional digital signature schemes such as the digital signature algorithm (DSA) or the elliptic curve digital signature algorithm (ECDSA) have signature sizes of at least 320 bits. The security analysis of adjusted TESLA starts with the possible attack models for breaking the one-way chain.

4.1 Attacks on the one-way chain

Attack model 1: Breaking the one-wayness:
Assume the one-way chain uses the one-way function $H:\{0,1\}^s \rightarrow \{0,1\}^s$ with $k_i=H(k_{i+1})$, and k_1 being the current public key. If a secure one-way function is used, the attacker has no other option to find a k_{i+1} for a given k_i than guessing k_{i+1}. If each k_i occurs with the same probability, the chance that an attacker guesses correct is $1/2^s$, so on average $2^s/2 = 2^{s-1}$ operations are needed. However, the used one-way function might not be completely secure. The used key needs to be secure for a long time. So even if a one-way function is chosen for which no attack is known today, a weakness in this function might be exposed in the future. Therefore, it might be possible to compute a k_{i+1} for a given k_i in $u<2^{s-1}$ operations. But computing one key k_{i+1} does not automatically mean that the system is broken. In our implementation, there will be one key for every second so that each key will only be valid for one second. If it takes the attacker one second or more to compute $k_2=H^{-1}(k_1)$, with $H^{-1}()$ denoting the inverse operation of the one-way function, the attacker has not gained any secret information, as k_2 will already be considered old. To be able to compute a key that is still valid, the attacker needs to be able to break the one-way function in less than a second. With a sufficient length of *s*, this will only be possible if a very big weakness is discovered in the used one-way function. If a well discussed one-way function, which has not shown any weakness yet, is used, this is very unlikely to happen. Hence, the biggest threat is attack model 2, the guessing-attack.

Attack model 2: Guessing-attack:
In this attack, the attacker tries to guess a key k' with $H^i(k')=k_1$ for some i, where k_1 is the current public key. The current public key is the last authenticated key by the receiver. If the output of the one-way function H() is random, an attacker can compute $k'_{i+1}=H(k'_i)$ until either $k'_{i+1} = k_1$ or $k'_{i+1}=k'_j$ for some $j \leq i$. In the case of $k'_{i+1}=k'_j$ for some $j \leq i$, the one-way function generated a circle, and the attacker needs to choose a new random number k'' as his seed guess. The attacker needs 2^{s-1} operations on average to find a key. Just like it is in the attack that breaks the one-way property, the attack will be useless if the attacker reveals a key which has already been published since he started the attack. However, in this attack the attacker does not just hash random numbers and compares them with k_1, but computes one-way chains himself. The chance that the one-way chain enters a circle is the same as the chance that a collision in a hash function occurs. If the one-way function can be seen as a random function, on average a collision occurs in $2^{s/2}$ operations, due to the birthday paradox. The chances that i is bigger than the length l of the one-way chain is very high, as $l<2^{s/2}$. In this case, the attacker would have broken the entire system, because the attacker would possess a one-way chain which is even longer than the original one-way chain.

To make this attack more difficult, a counter should be inserted into the one-way chain.(The proposed timestamp in section 3.2 can be seen as a counter.) If the one-way function $H:(\{0,1\}^s,\{0,1\}^m) \rightarrow \{0,1\}^s$ also includes a counter i, so that $k_i=H(k_{i+1},i)$, an attacker cannot simply apply the one-way function on the last computed value. If he wants to find a value k'_i with $k'_j=H(k'_{j+1},j)$ for $1 \leq j \leq i$ and $k'_1=k_1$, he needs to randomly pick a k'_i and compute

k'_1 for one previously chosen i. If k'_1 does not match k_1, he needs to guess another k'_i. To compute k'_1 the attacker needs i operations. Hence, on average the attacker will need $i \cdot 2^{s-1}$ operations. This is an increase in computation complexity in the order of i. It makes it much more unlikely that an attacker will find a k'_i with a big i. As the lifetime of the revealed key depends on i, a key with a small i will not last very long. If the used one-way chain has the length l, in the scenario without a counter, a successful attacker will, with a very high probability, receive a one-way chain of a length bigger than l. Thus the attacker will have broken the entire system and would only need to break the system once. But if a counter is used, the attacker will receive a one-way chain of the length i. As the complexity of the attack increases linear with the size of i, an attacker will only be able to break a one-way chain using a small i. Therefore, the attacker will only possess i keys, so that an attacker would only be able to break the system for a very limited time period. In our implementation for eLORAN, there is one key for every second, regardless whether or not a key will be published in this second. Therefore, a one-way chain of the size i will only last i seconds.

If it takes an attacker more than one second to compute 2^{s-1} operations, he will need $t>i$ seconds to compute $i \cdot 2^{s-1}$ operations and therefore calculate a key which is already expired. Hence, if the attacker cannot compute 2^{s-1} operations per second, the attacker needs to frequently update the public key k_1 which he wants to break with the last published value. This makes the attack non-deterministic and more complex as the search key needs to be updated frequently. On average 2^{s-1} operations are needed to find an s bit key. However, if after each try the search key is changed, on average 2^s operations are needed. Hence, on average $i \cdot 2^s$ operations are needed to find a valid key of the one-way chain, because the public key needs to be updated frequently if a counter is used.

Although, there will be one key for every second, not every key will be revealed. Depending on the message schedule, a key will probably only be revealed about every 60 seconds or more. If a key is only published every 60 seconds, a user that wants to verify a key needs to do 60 one-way function operations. Hence this increases the computation load of the user by the factor of 60. But it also increases the computation load of an attacker for the same amount. So from the security point of view this aggravates attacks on the one-way function as a larger i needs to be chosen. (It needs to be at least 60 to ensure that the attack reveals a fresh key.)

All this shows that adding a timestamp into the one-way chain generation process increases the security of the system significantly.

4.2 Key size

The authentication time and needed bandwidth for authentication purposes strongly depends on the used key size. To find out how long the key needs to be to provide sufficient security a model from Lenstra is used. The model is based on the widely accepted Moore's law. According to Moore's Law, the density of components per integrated circuit doubles every 18 months. A widely accepted interpretation of this law is that the computing power per chip doubles every 18 month [7]. Moore's law is not based on any physical law but only on the observation of the past developments. However, it has proven to be correct for 40 years now and it is believed to be a good assumption for the future.

Lenstra's model works as followed: At first an already breakable system, DES with a key length of 56 bits, is chosen as a reference. The security of DES is well studied and it is known quite well how secure DES has been for each year. The security level is chosen by deciding until which year the user trusted DES. Then Moore's law is used to predict until which year an s bit key will provide the same security as a 56 bit DES key in the reference year.

Lenstra's model is used to predict the security of symmetric block ciphers, for which no other attack as brute force (testing all possible keys) is known. However, we do not want to know the key length of a block cipher but that of an adjusted TESLA one-way chain. In section 4.1 we showed that breaking an s bit adjusted TESLA one-way chain requires even more than s tries, due to the embedded timestamp. Hence, breaking an s bit one-way chain is even more complex than breaking an s bit block cipher. In our analysis we considered this and analyzed how secure a one-way chain generated with 56-bit DES would have been in 2006. (Block ciphers can be used to efficiently build one-way chains and our implementation of the one-way chain is based on the AES block cipher)

Three parameters determine the complexity of an attack on a one-way chain with an embedded timestamp:

- The parameter i defines the number of keys the attack reveals and therefore the time period during which the attacker can spoof a receiver.
- The average attack time
- The costs of the attack

As discussed in section 4.1, to break an s bit one-way chain for i seconds, $i \cdot 2^s$ tries are needed in average, while only 2^{s-1} tries are needed in average to break a symmetric block cipher with s bits. This significantly increases the complexity of an attack on the one-way chain. We made the security analysis for different values for i. If the attacker only reveals keys that are valid for a short time period such as one minute, it is unreasonable to believe

that an attacker accepts attack times of several days. On the other hand if the attacker reveals keys that last a day, the attacker might be willing to accept an attack time of as much as a month or more. Figure 4 shows the attack time and cost for three different *i*. To consider how small *i* can be for an attacker to be still useful, several things need to be considered.

If *i*=1 is chosen, the attacker has one second to mount the attack after a correct key was found, minus the time it took to compute the key. This scenario is impossible, as the receiver needs to synchronize with the spoofed signal first. If a MAC is used, the attacker would also not be able to create a valid MAC, as the attacker needs to know the key *d* seconds ahead of time to create a valid MAC, where *d* denotes the minimum delay between the MAC and the corresponding key. If the authentication bandwidth is 7.5% and the next key is the authentication key, *d* is 32s. But there are also other factors that increase the needed size of i that also apply to the case that adjusted TESLA without a MAC is used.
The attack will not be able to mount the attack immediately after a key is found. In practice, the attacker needs to transmit the key to the spoofing device and the spoofing device needs to take control over the LORAN data channel before this attack can be started. Furthermore, due to the message schedule, the attacker might need to wait until the receiver accepts another key as the receiver might expect correction messages first. We assume that the attacker needs at least about 30-60 seconds to start the attack. Another factor that increases the complexity of the attack is the fact that an attacker will not be able to update the search key every second. Although there will be a key for every second, only about every minute a key will be released. Therefore, the attacker will only be able to update a key every 60 seconds or more depending on the key schedule. Therefore, to be able to compute keys that last for at least 60 seconds an attacker needs to choose i=60s+60s=120s.

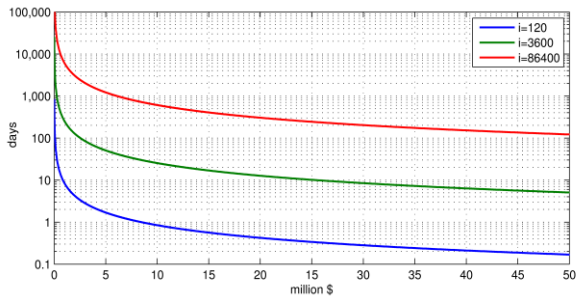

Figure 4: The average attack time in days to find i keys with a key size of 56 bit for i=120 (keys for 1-2 minutes), i=3600 (keys for one hour) and i=86400 (keys for one day) in 2006.

To find 120 keys in one day 9.2 million dollar were needed in 2006. With a budget of 1 million dollar the attack would have taken about 9.2 days in 2006. During all this time the attacker would need to be ready to start the attack within a minute and will only be able to attack about one one-way chain key.(in case the authentication bandwidth is about 7.5%) To attack a transmitter station for one hour 10 million dollar and about 27 days are needed. To generate keys that are valid for one day 50 million dollar for an average attack time of 132 days are needed in 2006.

In these attacks only keys for one one-way chain are found. Finding keys for two one-way chains at the same time is much more difficult and much more resources would be needed. However, breaking keys for only one one-way chain limits the attacker a lot. The attacker would only be able to generate signals of one transmitter station. A signal synthesis attack would not be possible, as at least three signals from different transmitter stations are needed to determine the position.

Hence attacking a 56 bit adjusted TESLA one-way chain in 2006 would have been very complex. The attacker would have needed a lot of resources, while he would only be able to break the system temporarily. Furthermore, attacking more than one chain at the same time would have required even much more resources. Therefore, we assume that a 56 bit one-way chain was reasonably secure in 2006, so that we choose 2006 as our reference year.

According to Lenstra's model and Moore's Law, the attacks described for a 56 bit one-way chain in 2006 will not be possible until about the year 2037. Table 1 lists different key lengths and the years until which these key lengths provide a security equivalent to a 56 bit chain in 2006. For a more detailed security analysis see [1].

key length	year	key length	year
75	2030	120	2089
80	2037	130	2102
90	2050	140	2115
100	2063	150	2128
110	2076	160	2141

Table 1: The key lengths and years which provide equivalent security as 56 bit key provided in the year 2006.

If message authentication is necessary and a MAC is used, this MAC can be quite short. A MAC length of only 32 bits will provide enough security to ensure that an attacker cannot forge a message. Each key is only used to generate one MAC. And the only entity that knows the key before it is published is the transmitter station. When

an attacker forges a MAC, he cannot check if his forged MAC is valid without the knowledge of the key. Therefore, the attacker needs to send this forged MAC to the victim without testing it. This results in a very limited number of tries the attacker can perform during one time interval. However, in average the attacker needs 2^{32} tries if the MAC size is 32 bits. A more detailed analysis of the needed MAC size can be found in [1].

5.0 IMPLEMENTATION OF ADJUSTED TESLA

One big problem of TESLA is the distribution of the public keys. In TESLA, small one-way chains are used that are only used for a short time period. For each chain the public key is authenticated using a digital signature. However, this approach is very inefficient for eLORAN. Digital signatures have sizes of several hundred bits. It would be necessary to frequently transmit these digital signatures, as users might arrive in the coverage area of a LORAN transmitter at any time. These users would need to wait until the digital signature is repeated before they can authenticate the signals from this station. All this revokes the advantages of TESLA compared to digital signatures. The authentication bandwidth as well as the authentication time would increase significantly. Therefore, in adjusted TESLA, we propose to use only one one-way chain for each station for several years. In this way the public key can be implemented into the receiver and no digital signature needs to be transmitted over the LORAN data channel. However, to do this it must be possible for the receiver to validate keys even if the public key is several years old. Assume that the public key is one year old and there is one key for every second, then approximately 2^{24} nodes of the one-way chain need to be computed to validate the key. We chose a one-way chain based on AES to achieve the needed performance. With a hardware implementation of AES it is possible to compute 2^{24} keys in about one second. So validating a key with a public key that is 30 years old will only take about have a minute. To avoid computing this many nodes for every key, the public key should be frequently updated with the last published key. In this way the receiver only needs to do such a large number of computations in case that the receiver has not been in the coverage area of this particular transmitter station for years. Once the public key has been updated, the validation can be done within microseconds or even nanoseconds.

One way to avoid a situation where a receiver possesses a public key for a station that is several years old, the system could send out current keys for all station in the almanac data. The keys are only 80 bits and no digital signature is needed to authenticate them. It is also not necessary to send them out frequently, because the key updates are only used to prevent the receiver from holding public keys that are many years old since the validation of these keys may take up several seconds. Hence, sending these keys in the almanac data would not increase the needed bandwidth a lot, as even sending one data packet each hour to transmit the keys would probably be enough to ensure that eLORAN users would always possess public keys for each station that are newer than one year. It is also possible to update the public keys over a second channel such as the Internet.

However, keeping in mind that the authentication time of a LORAN signal will be several seconds due to the low data rate, it is an acceptable assumption that the validation of the first key of a transmitter station that was not received for years takes several seconds as well. Even if the keys are not updated by almanac data or a second channel, the performance of adjusted TESLA is still better than other authentication mechanism in the worst case. In the average and best case adjusted TESLA will far outreach other authentication mechanism in authentication time.

Figure 5 illustrates the generation of the adjusted TESLA one-way chain with the use of AES. A more detailed description of our implementation can be found in [1].

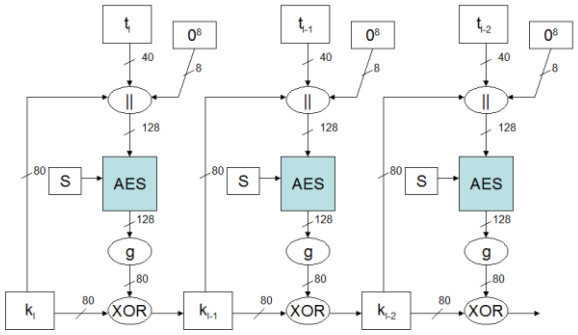

Figure 5: The generation of an adjusted TESLA one-way chain based on AES. K_i denotes the one-way keys, S a public 128 bit string that is used as the key during the AES encryption operation. S is unique for each one-way chain. t_i denotes a timestamp, || the concatenation of two strings, AES the 128 bit encryption of AES with the key S, g denotes a function that reduces the 128 bit input to an 80 bit output and XOR denotes the exclusive-or operation.

The authentication time of TESLA mainly depends on the available bandwidth for authentication purposes. The transmission of one data packet containing 41 bits of payload takes up to 2.4 seconds. To transmit an 80 bit adjusted TESLA key two messages are needed. If a 32 bit Mac is used, a third message is needed. However, the LORAN data channel is also used to transmit time, correction and other messages. Therefore only a fraction of the data bandwidth will be available for authentication purposes. We estimate that a bandwidth of roughly 7.5% for authentication purposes is realistic. But this strongly depends on how much bandwidth the other LORAN

messages need and whether or not another data channel is implemented in eLORAN [15].

Table 2 lists the authentication time in dependence of the available bandwidth. One big disadvantage of the original TESLA is that the MAC can only be authenticated after some delay. However, because the transmission of one data packet takes already 2.4 seconds and because other messages are also transmitted between authentication messages, the keys can be send directly after the MACs. The security delay will still be long enough so that this disadvantage of TESLA does not count in adjusted TESLA.

	Only an 80 bit key	80 bit key with an 40 bit MAC		
band-width	authentication period	authentication period	authentication delay	security delay
100 %	4.8 s	7.2 s	7.2 s	2.4
50 %	9.6 s	14.4 s	12 s	4,8s
10 %	48 s	72 s	50.4 s	24 s
7.5 %	64 s	96 s	66.4 s	32 s
5 %	96 s	144 s	98,4 s	48 s
2.5 %	192 s	288 s	194,4 s	96 s
1 %	480 s	720 s	482,4 s	240 s

Table 2: The authentication time in dependence of the available bandwidth for authentication purposes for adjusted TESLA. Authentication period is the time it takes after one message is authenticated until the next message is authenticated. Authentication delay is the time it takes after the transmission of a message until this message is authenticated. The security delay is the maximal allowed offset between the receiver and transmitter clock.

6.0 CONCLUSION

In this paper we introduced an efficient authentication mechanism for eLORAN, called adjusted TESLA. This authentication mechanism can be used to increase the security of navigation systems against signal-synthesis attacks as well as counterfeit correction message attacks. The main innovation in this paper is the efficiency how this is achieved. We showed that the very powerful signal-synthesis attacks can be prevented with the use of only an 80 bit adjusted TESLA key. With an additional 32 bit MAC, counterfeit correction messages can be prevented as well. However, this MAC is optional, as the keys on their own already improve the security of eLORAN and other navigation systems significantly. Due to the small key length and the fact that the MACs can be released directly after the keys, the authentication time is significantly reduced as well. Compared to DSA or ECDSA signatures that are 320 bits long or the first proof-of-concept implementation of TESLA in eLORAN [4] that needs 320 bits, our implementation only needs 120 bits (80 bit key plus a 40 bit MAC), which reduces the authentication time by 62.4%. If no MAC is used, the time to authenticate the signal is even reduced by at least 75%. Furthermore, adjusted TESLA provides better long-term security, as breaking an 80 bit adjusted TESLA one-way chain is more difficult than breaking a 320 bit DSA or ECDSA signatures. Therefore, an 80 bit adjusted TESLA one-way chain will be reasonable secure for approximately at least another 20 years, while the use of a 320 bit digital signatures is only advisable until 2013 [11].

Hence, adjusted TESLA is an efficient authentication mechanism for navigation systems, especially if only a small data rate is available for authentication. However, message authentication is only the first step towards a secure navigation system. To prevent the powerful selective-delay attacks or relaying attacks further countermeasures are needed. For GNSS systems hidden markers is a possible solution. Unfortunately, hidden markers cannot be applied to terrestrial navigation systems such as eLORAN. New ideas like colliding signals [1] are needed to solve this problem for terrestrial navigation systems.
Nevertheless, eLORAN with adjusted TESLA would not only be a backup for current GNSS systems, but would be a real alternative to current civil GNSS systems for application that require the highest possible security level against attacks.

REFERENCES

[1] G. T. Becker, 2009, “Security mechanisms for positioning systems - enhancing the security of eLoran”, master thesis, Ruhr-University Bochum
[2] L. Scott, 2003, “Anti-Spoofing & Authenticated Signal Architectures for Civil Navigation Systems”, Proc. ION GPS/GNSS, pgs. 1543-1552.
[3]M.G. Kuhn, 2004, “An Asymmetric Security Mechanism for Navigation Signals”, Proc. Information Hiding Intl. Workshop, pgs. 239-252.
[4 D. Qiu, 2007, “Security Analysis of Geoencryption: A Case Study Using Loran”, Proc. ION GNSS, pgs. 1146-1154.
[5] C. Wullems, O. Pozzobon, and K. Kubik, 2005, “Signal Authentication and Integrity Schemes for Next Generation Global Navigation Satellite Systems”, Proc. ENC GNSS 2005.
[6] S. Lo, B. Peterson and P. Enge, 2009 “Assessing the Security of a Navigation System: A Case Study using Enhanced Loran”, Proc. ENC-GNSS 2009
[7] A. K. Lenstra, 2006, “Key lengths”, In Handbook of Information Security, Volume1: Key Concepts,

Infrastructure, Standards and Protocols. John Wiley & Sons.
[8] A.K. Lenstra and E. R. Verheul, 2001, “Selecting Cryptographic Key Sizes”. Journal of Cryptology: the journal of the International Association for Cryptologic Research, 14(4):255–293.
[9] T. Güneysu, T. Kasper, M. Novotny, C. Paar and A. Rupp, 2008 “Cryptanalysis with COPACOBANA”.,IEEE Trans. Computers, 57(11):1498–1513.
[10] S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, and M. Schimmler, 2006, “Breaking Ciphers with COPACOBANA - A Cost-Optimized Parallel Code Breaker”, In Eighth Intl Workshop Cryptographic Hardware and Embedded Systems (CHES 06), pages 101–118. Springer,
[11] E. Barker, W. Barker, W. Burr, W. Polk, and M.Smid, 2007, “NIST SP800-57: Recommendation for Key Management Part 1: General(Revised)”, Technical report.
[12] T. Humphreys, B. Ledvina, M. Psiaki, B. O’Hanlon, and P. Kintner, 2009, “Assessing the Spoofing Threat”, GPS World, pages 28–38, Jan 2009.
[13] Sherman Lo, Benjamin Peterson, and Per Enge, 2008 “Using Loran for Broadcast of Integrity Information for Modernized Global Navigation Satellite Systems (GNSS)”. In Proc. ENC-GNSS, 2008.
[14] B. Forssell, “The Dangers of GPS/GNSS”, Coordinates, February 2009.
[15] Peterson, Benjamin, “Feasibility of Increasing Loran Data Capacity using a Modulated Tenth Pulse”, Proceedings of the International Loran Association 36th Annual Meeting, Orlando, FL, October 2007