Authenticating Aviation Augmentation System Broadcasts

Sherman C. Lo, *Stanford University*
Per K. Enge, *Stanford University*

BIOGRAPHY

Sherman C. Lo is currently 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 alternate navigation.

Per Enge is a professor in the Department of Aeronautics and Astronautics at Stanford University. He is the director of the Stanford GPS Laboratory and the Center for Position, Navigation and Time.

INTRODUCTION

An important function of augmentation systems for Global Navigation Satellite Systems (GNSS) is providing information need to guarantee the integrity of GNSS derived position, navigation and time (PNT) outputs. This mission is the primary purpose of spaced based and ground based augmentation system - SBAS and GBAS, respectively. These systems are designed to serve aviation navigation and landing by providing information needed to assure safe use of GNSS. Thus, information integrity is fundamental to these augmentation systems. One component of information integrity is the ability to authenticate the source of the data. While this assurance is currently not built into these systems, it may be possible to overlay authentication capability.

Traditional data authentication techniques can be used to provide source assurance. However, augmentation systems have requirements that differ from the channels for which these techniques were designed. In particular, the data is more time sensitive and the bandwidth is much more limited. Additionally, user and system equipment are designed for decades of service with little to no upgrades. As a result, aviation seeks data authentication that is 1) fast, 2) robust to message loss, 3) not resource intensive 4) self contained and 5) robust to future attacks. Traditional data authentication techniques must be adapted to achieve these targets with limited bandwidth and limited two way communications.

Meeting these desired qualities may be difficult given design constraints imposed by low bandwidth, avionics, and airspace infrastructure. However, the characteristics of augmentation systems and its operations may also aid the design. These attributes limit the types of attacks that are feasible against the system as well as provide means to cross check information.

The paper starts by examining the reasons for and desirable features of authentication on aviation augmentation systems. Next, it considers basic cryptography and traditional data authentication techniques suitable for the aviation broadcast environment. Protocols based on asymmetric and symmetric key are discussed. Additionally, key strength and related issues are looked at. It then examines the important consideration of key distribution as this may be a major hurdle to adoption. This paper presents a key distribution protocol that utilizes the operation of the aircraft and air traffic to aid in key verification. The last section of the paper presents some case study designs for SBAS and GBAS. These designs are not meant to be proposal but rather to give some idea about feasibility and data requirement.

MOTIVATION & GOALS

Communication navigation and surveillance (CNS) in civil aviation is moving towards predominantly digital data centric architectures. Aviation augmentation systems such as SBAS and GBAS are tasked with providing navigation integrity. It follows that data integrity, perhaps in the form of authentication, is a useful and logical development for these systems. In fact, data or source authentication was proposed when the SBAS concept was being developed.

Data authentication of augmentation systems is a useful first case study for developing and implementing enhanced information security for the national airspace (NAS). Assessing and developing security for these systems can provide useful insights, understanding and familiarity. At the same time, it is a closed system with a limited scope in terms of problems and possible attacks. With augmentation systems, the goal is solely data authentication rather than more complicated tasks such as location authentication. Security in other systems may rely on multiple systems or systems beyond the control of aviation. Additionally, the characteristics and operations of augmentation system limit the scope of possible attacks.

AUTHENTICATION ON AUGMENTATION SYSTEMS

Providing data authentication for augmentation system is a timely concept. A form of authentication is being considered in the GBAS VHF data broadcast (VDB) proposal for Category II/III approach and landing [1]. The idea, shown in Figure 1 uses human in the loop verification to ensure that the VDB data time slot used for receiving GBAS information corresponds with that listed on the approach plate. The VDB uses one of eight available time slots in a frame as seen in Figure 2, leaving the rest potentially unused. A spoofer could broadcast on an unused slot. As broadcast informs the avionics which slot to receive, a receiving a spoofed broadcast will cause the avionics lock on to an incorrect slot and continue using its transmissions. The proposal thus prevents this deliberate deception of the GBAS avionics. While the proposal addresses a specific data spoofing vulnerability, it does not prevent other means of data spoofing such as overpowering or disabling and replacing the legitimate VDB broadcast. Cryptographic data authentication provides a more general method that can prevent this form of attack.

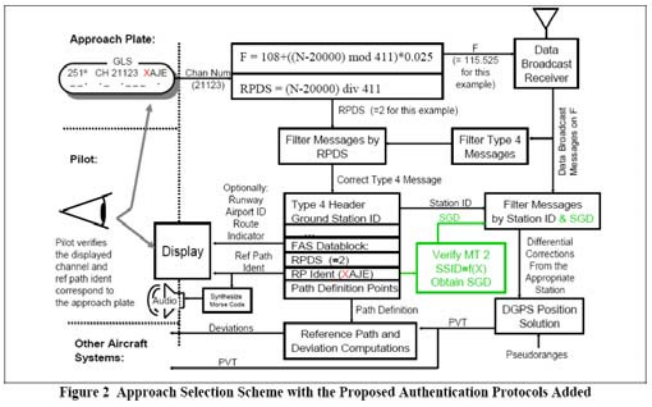
Approach Plate:
GLS
251° CH 21123 XAJE
Chan Num (21123)
F = 108+((N-20000) mod 411)*0.025
RPDS = (N-20000) div 411
F
(= 115.525 for this example)
Data Broadcast Receiver
RPDS (=2 for this example)
Pilot:
Filter Messages by RPDS
Filter Type 4 Messages
Data Broadcast Messages on F
Correct Type 4 Message
Optionally: Runway Airport ID Route Indicator
Type 4 Header Ground Station ID
Station ID
SGD
Filter Messages by Station ID & SGD
Pilot verifies the displayed channel and ref path ident correspond to the approach plate
Display
Ref Path Ident
FAS Datablock:
RPDS (=2)
RP Ident (XAJE)
Path Definition Points
Verify MT 2 SSID=f(X) Obtain SGD
Differential Corrections From the Appropriate Station
Audio
Synthesize Morse Code
Path Definition
DGPS Position Solution
Deviations
PVT
Reference Path and Deviation Computations
Other Aircraft Systems:
PVT
Pseudoranges
Figure 2 Approach Selection Scheme with the Proposed Authentication Protocols Added

Figure 1. Proposed Authentication for CAT II/III GBAS [1]

Source or data authentication on SBAS was studied as the system was being conceptualized. One objection is it takes a significant amount of data that it would require. Another is the infrastructure needs for authentication. The purpose of this paper is to re-examine these areas and determine how to ameliorate these objections.

SBAS does ensure data accuracy against errors (not spoofing) through parity check and error correction. Additionally, SBAS has a redundancy of sources with a goal of coverage by at least two geostationary satellites over its service volume.

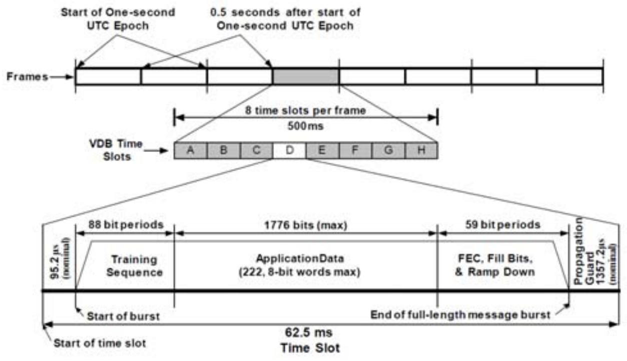
Start of One-second UTC Epoch
0.5 seconds after start of One-second UTC Epoch
Frames
8 time slots per frame
500ms
VDB Time Slots
A
B
C
D
E
F
G
H
88 bit periods
1776 bits (max)
59 bit periods
95.2 μs (nominal)
Training Sequence
ApplicationData (222, 8-bit words max)
FEC, Fill Bits, & Ramp Down
Propagation Guard 1357.2μs (nominal)
Start of burst
End of full-length message burst
62.5 ms Time Slot
Start of time slot

Figure 2. LAAS Frame and Data Structure [2]

DESIRABLE QUALITIES

We start by developing an understanding of features that are desirable for authentication on aviation augmentation systems. These features are driven by the characteristics of the systems. First, data on these systems is very time critical and needs to be used on the order of seconds. Furthermore, the data channel is generally very bandwidth constrained. Second, equipment is also limited. User avionics is not networked and, once installed, is expected to operate for many years with few major changes. Additionally, increased complexity can greatly increase certification costs. Service provider equipment is also very slow to change. Given the life cycle of aviation systems and equipment, any system changes typically needs to be backwards compatible.

Aviation augmentation systems thus seeks data authentication that is 1) fast, 2) message loss tolerant, 3) not resource intensive 4) self contained and 5) robust to attacks 20 or more years in the future. Traditional data authentication techniques must be adapted to perform to these specifications under limited bandwidth conditions.

DATA AUTHENTICATION TECHNIQUES

This paper examines basic, traditional cryptographic techniques for data authentication. Detailed descriptions are found in security books such as [3][4]. More recently developed techniques may also prove useful and enhance performance. However, this is beyond the scope of the paper.

BASIC CRYPTOGRAPHIC KEY DESIGNS

Cryptographic authentication can be achieved using either public (asymmetric) or symmetric key. In public key cryptography, a public and private key pair is used. The private key is kept by the sender and can be used to digitally sign a message hash for the purpose of authentication. The public key is freely available and can be used to verify the signature and derive the hash. Only a holder of the private key can produce a valid signature. The hash then verifies that the message has not been tampered with. Hence public key can verify the message and its sender. With symmetric key, the same key is used by both the sender and the receiver. With specific protocol designs such as timed efficient stream loss-tolerant authentication (TESLA), properties similar to asymmetric authentication can be achieved though with additional requirements. The advantage in using symmetric keys is that they are much more data efficient (at least 2 times but can be much more) and computationally faster (100 or more times) than asymmetric protocols. These benefits are particularly relevant for aviation. Aviation typically has low bandwidth channels (100s of bps) while its data is highly time sensitive. A comparison of the required key sizes for different security levels for symmetric and some forms of asymmetric keys is given in Table 1.

Symmetric Key size (bits)	Asymmetric (RSA & Diffie Hellman) Key Size (bits)	Asymmetric (Elliptic Curve) Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Table 1. NIST Recommended Key Sizes (Each row has roughly the same security level) [5]

TRADITIONAL DATA AUTHENTICATION

Digital signatures and digitally signed hash are public key based methods for verifying for data authentication. A generalized concept is shown in Figure 3. The basic idea is for the sender to use a cryptographic hash function to take convert the message bits into a fixed length value. The hash function should have certain properties such as ease of calculation, one-wayness, and being collision resistant. One-wayness means that the hash can be calculated from the message but not vice versa. Collision resistant means two messages are very unlikely to result in the same hash. The sender uses their private key to “sign” or “encrypt” the hash. The receiver uses the public key to recover the hash and compares it to the hash generated using the received message. If the two hashes match, then the message integrity and source is verified. This is because only the holder of the private key can generate a signed hash that is decodable into the message hash using the public key. As only the holder of the private key can generate the hash, the message content cannot be repudiated. Use of public key encryption typically requires a trusted third party or certificate authority (CA) to provide a certificate attesting to the authenticity of the public key. Typically, this certificate contains information to determine authenticity including the key originator and the signature of the authority. As a result, it can require tens or hundreds of bytes of data. Additionally, there should be a means of revoking a key that has expired or been compromised. The CA often plays a major part in key revocation.

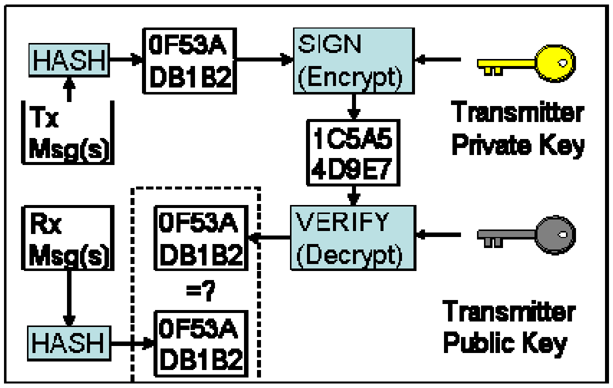

Figure 3. Authentication using Public Key Cryptography

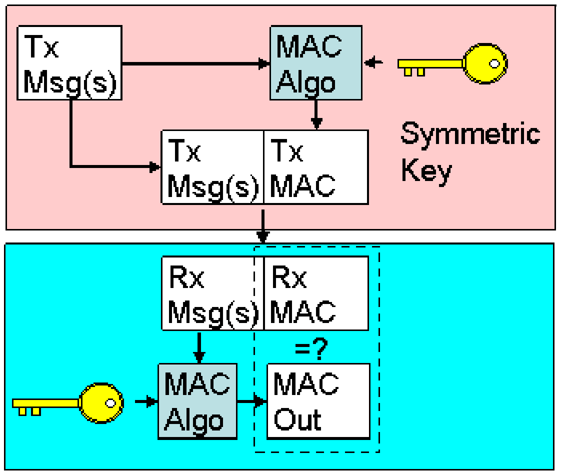

Figure 4. Authentication using Symmetric Key Cryptography

A standard for public key authentication is digital signature algorithm (DSA) and its extension known as elliptic curve DSA (ECDSA). ECDSA improves upon DSA performance by using elliptic curve cryptography (ECC). ECDSA reduces the data requirement and improves computational efficiency for a given security level. The improvement can be seen in Table 1. One concern with the adoption of elliptic curve based cryptography is patent issues. However, the use of ECC for applications such as safety and security of the NAS, may be covered by the National Security Agency intellectual property license.

The traditional means of authentication using symmetric encryption is with message authentication code (MAC) or keyed hash function. The MAC is a set of data derived from a message for the purpose of authenticating that message. Figure 4 shows the basic idea where the source transmits the message with a MAC generated using a MAC algorithm, the message and a symmetric key. The MAC algorithm may be hash based or otherwise. The recipient verifies the message by performing the same algorithm with the same key on the received message. The recipient then compares the resultant MAC with the received one. Again, this provides simultaneous verification of the data integrity and source authenticity to holders to the symmetric key at the time of transmission. However, this technique suffers from using the same key for sender and recipient (and thus potential spoofers). So, unlike a public key, the symmetric key must remain secret. This is not very suitable in a broadcast environment when any one can have access to the key.

TESLA

So to use symmetric encryption for authenticating augmentation systems, one must create asymmetry so that spoofers cannot generate messages that may be accepted as valid. TESLA developed for packetized data [6], is one protocol that has been suggested by numerous parties for navigation authentication [7][8][9]. Figure 5 illustrates the concept of TESLA. Basically, it works on the principle of delayed release of the authentication key. TESLA is set up by having the sender generate a secret key K_N and creating a chain of keys from it using one way hashes (F) as follows, where K_{N-n} is an intermediate TESLA key:

$$K_{N-n} = F^n\left(K_N\right)$$

with $F^n()$ is the operation of the function F, n consecutive times. The last key in the chain is K_o which we will term the base TESLA key. This key is distributed via a trusted and preferably secure means to the user some time prior to using authentication. The authenticity of K_o needs to be assured, perhaps with by a CA.

One can think of transmissions are being segmented in multiple intervals. In each interval n, the message or messages, M_n, a message authentication code (MAC), MAC_n , and a key K'_{n-i} (valid for a prior interval) are transmitted. In TESLA, the MAC is generated from the message(s) and a secret key. One means is to use a keyed hash MAC (HMAC) where a key and a hash function are used to generate the MAC. In the figure, the MAC generation key for interval n is denoted by K'_n. The MAC generation key K'_n is generated from a one way hash (F', a different hash function than F) of the current intermediate TESLA key K_n. Note that prior to broadcast of K_n, the both K_n and K'_n are only known to the sender. Thus, only the sender can generate the MAC. K_n is later broadcast at time t in a later interval $n+i$. In the figure, i =1.

Verification comes in two steps. After receiving K_n, the receiver can derive the MAC generation key, $K'_n = F'\left(K_n\right)$, and verify that the MAC was generated from the message and the derived MAC generation key. Assume that the user is loosely synchronized with maximum error, e and knows the key transmission schedule. Then messages and MACs (based on key K'_n) received prior time t-e (as measured by the user) can only be generated by the authentic sender. Any message with MAC based on K'_n received afterwards is considered suspect. The sender will have already started using the next keys in the chain K_{n+j}, j > 0 and enabling the verification continues uninterrupted

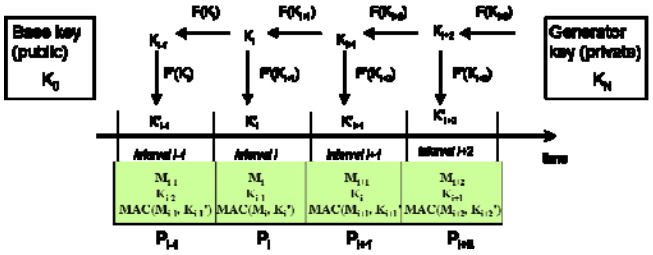

Figure 5. TESLA with key released delayed by 1 interval

The step above provides verification that the message was unaltered and derived from the holder of K_n. The next step is to validate that the key is from the legitimate source. In TESLA, the validation is done using the base TESLA key from our legitimate source, K_o. It can be verified that both K_o and K_n are derived from K_N provide the following are the same:

$$F^n\left(K_n\right) \leftrightarrow K_0$$

Since we know K_o is from the trusted source, it follows that K_n is from the legitimate source. K_o thus is used to tie our the key used for generating the MACs to the trusted source. Given it creation, it cannot be used to generate those keys but can be used to verify them. As long as the provenance of K_o is good, the whole chain can be verified. The secure distribution requirement is more reasonable because, unlike the previous keys, base TESLA key does not need to be updated as frequently.

However, the standard implementation of TESLA may not be for a low bandwidth channel such as SBAS. First, we need to solve the key distribution issue for the base key. Second, we need to be bandwidth efficient.

KEY DISTRIBUTION

Cryptography based data authentication will require key distribution. For signed hash, the public key needs to be provided and its source guaranteed. This typically requires some sort of certification or public key infrastructure (PKI). For the avionics, the key may be loaded and validated prior to installation via a network connection or preloaded when built. This is necessary as the receiver may never be networked once it has been installed in an aircraft. However, this does not accommodate key revocation or the need to change public/private keys. For TESLA, a new base TESLA key needs to be provided on a regular basis as a result and its provenance too needs to be assured.

Key management issues such distribution and revocation will be discussed a later section.

MODIFYING TESLA KEY USE

Incorporating TESLA into a constrained data channel such as SBAS may necessitate modifying the algorithm to reduce its bandwidth requirements. We propose modifying the TESLA algorithm whereby hashed keys are sent less frequently then MAC. That is, we hash multiple messages with the same key, extending the use of the key. This is usually not recommended as increases vulnerability. However, given the short amount of time and limited number of messages that the key will be used, the choice seems acceptable. The SBAS case study will illustrate an implementation of this concept.

REASONABLE THREATS & ATTACKS

In addressing data security for aviation augmentation system, it is important to understand what the vulnerabilities are and which we are addressing. In this paper, the primary threat of concern is on-air spoofing from remote spoofers. Other threats can and should be managed by other means. Data security issues related to the upload and broadcast of message within the augmentation system are matters of physical security and beyond our scope. Similarly, security against onboard local spoofing and injection spoofing/simulation should also be solved through physical security. The former, termed a "limpet spoofer" by Scott, is a device is placed aboard the vehicle of interest to broadcast signals that only affect that vehicle. The later is the introduction of an injected signal into the RF input of a receiver, thus bypassing the antenna.

The cryptographic strength of the authentication depends on the type of attack that can be made on it. Data authentication systems in cryptography have to be robust against a variety of different attacks. A basic attack is the brute force attack whereby the attacker tries to determine the authentication key by trying all possibilities. More sophisticated attacks are possible. In understanding the authentication strength needed, we need to be able to determine both the attacks that can be made and how long it would take an attacker to defeat the authentication such that they can generate messages that will be accepted by the user.

Start by examining the possible attacks. This is where the simplicity of the augmentation system works in our favor. First, these are broadcast systems whose inputs are not influenced by forces outside the system (besides GNSS measurements). This means an attacker, unless there is an insider, cannot perform a chosen message attack where they get to ask the system to send some number of selected messages to be authenticated. Second, the collision attacks where the attacker only has to find different messages with the same hash are not useful. This is because the attacker does not have the key used to generate the hash and thus cannot create their own messages. Instead, they must listen to the broadcast for messages. For an 80 bit hash, one needs to get 2^{40} messages (due to the "birthday problem" [4]) to expect find a repeated hash. This is reasonably trivial if the attacker can generate the message but if the attacker has to listen to SBAS or GBAS, assuming one message hash a second, they would have to listen for 35000 years.

Another factor in our favor is that not all possible messages are valid or useful for spoofing. A limited number of messages are valid because the internal data need to be consistent with possible message types, cyclic redundancy code (CRC), etc.

Time to break depends on the algorithm used and several other factors. As an illustrative example, we examine HMAC as they are employed for TESLA. In HMAC, a key (*K*), the desired message (*M*) and a cryptographic hash function (*F*) are used to generate the MAC [10].

$$MAC = F\left(M, K^{'}\right)$$

The strength of the MAC depends on both the hash function and the key size. Common MAC functions used and their output hash lengths are seen in Table 2. Some of these hash functions have known vulnerabilities which weaken them to certain attacks. However, these vulnerabilities are not necessarily applicable when using them for HMAC. Bellare, et. al. states that key used should be at about the length of the hash output with longer keys not being significantly more secure and shorter keys being less secure [10]. Additionally, the full output may not be necessary though it is suggested that one should not use less than half of the output bits.

The attacker will find it useful to attack one of two mechanisms. First, it can try to determine K_n before it is revealed so that fraudulent messages can be made. To do this, it knows the previous K_{n-i}, ($i <= n$), the hash functions *F*, *F'*, and messages with MAC generated from $K_n^{'} = F\left(K_n\right)$. Being able to determine K_n allows the spoofer to transmit false messages until the true K_n is revealed. As it is expected that a new K_n is used every 20 to 60 seconds, this only provides a short window of time to break the key and spoof. The second, more valuable attack is to try to determine K_N given same knowledge above. As K_N is used to generate the entire chain of keys, it is more valuable and has a longer utility (*N* times that of each key in the sequence). Hence, the key length is driven by the security requirements and life time of K_N.

Hash Function	Output Length (bits)
MD4	128
MD5	128
SHA1	160
SHA2 (SHA-256/224)	256/224

Table 2. Common cryptographic hash functions and bits required

Given an attack, we can estimate how long it will take an attacker with to discover the key. The results are seen in Table 3 assuming brute force attack. The time to break values and equipment assumptions are based on [3]. [3] gives the time to break for \$1 M and \$1 B of hardware (in 1995). Assuming the "Moore's Law" rule of thumb whereby transistors on an average integrated circuit (the inverse of computation cost) doubles every 18 months, this is roughly equivalent to \$1000 and \$1 M of 2010 hardware, respectively.

Symmetric key length (bits)	Time to break (\$1 K in 2010)	Time to break (\$1 M in 2010)
80	7000 years	7 years
128	10^{18} years	10^{15} years
160	$4x10^{27}$ years	$4x10^{24}$ years
192	$2x10^{37.3}$ years	$2x10^{34}$ years
256	$3x10^{56}$ years	$3x10^{53}$ years

Table 3. Hash Effective Strength in bits and time to break for Brute Force Attacks using on \$1 K and \$1 M Hardware in 2010 [3]

Given the long life cycle of avionics and aviation systems, understanding how time to break will be affected in the future is a critical element. In designing the authentication scheme, we must choose one that has adequate strength towards the end of life which may be 20-30 years in the future. One can estimate the effect of gradual increases in computational capability. Again, we can use "Moore's Law" to approximate the effect. Table 4 shows the result for a doubling every 18 months. Without other vulnerabilities, somewhere between 80 to 128 bits is adequate. However, vulnerabilities which can significantly reduce the strength of the hash algorithms have been found. Though some of these vulnerabilities may not be applicable to HMAC, one should be mindful of the possibility of future discoveries and plan accordingly. Hence, having the equivalent of a symmetric key of at least 160 bits seems prudent.

Years from 2010	80 bit	128 bit	160 bit
0	7 years	10^{15} years	$4x10^{24}$ years
12	10 days	$4x10^{12}$ years	$1.6x10^{22}$ years
24	1 hour	$1.6x10^{10}$ years	$6.3x10^{19}$ years
36	13 seconds	$6.3x10^{7}$ years	$2.5x10^{17}$ years

Table 4. Table of symmetric key strength vs. time to break with \$1 M equipment & Moore's Law

KEY DISTRIBUTION

Key distribution is a significant issue for augmentation systems because of several constraints. One constraint is the limited bandwidth. Public keys require more bandwidth to distribute. While shorter keys may be used, these need to be updated more frequently as they can be cracked in less time. Bandwidth constraints affect symmetric key based algorithms such as TESLA as well. In TESLA, the time to authenticate depends on the time between the broadcast a message and the current TESLA key (i.e., K_n) needed to validate the MAC of the message. The more frequent broadcast of the intermediate TESLA key requires more bandwidth. The bandwidth thus constrains our ability to distribute key of the desired strength in an acceptable time frame.

Another constraint is that there is no aircraft to ground network that allows for verification the distributed key. This constrains the ability of the aircraft to securely verify received keys with a trusted certificate authority.

We offer a couple of key update strategies address these limitations that leverage some of the characteristics of aviation One way is to use FAA chart update schedule (56 days) and distribute keys along with these publications. For commercial aviation, this may be reasonable as these electronic updates of charts are routine. It overcomes bandwidth issues and some trust issues since you have to trust the source of your aviation charts. However, it is vulnerable to social engineering and insider attacks. Another drawback is that there is no high integrity mechanism to directly input such information to the navigation system.

Another idea is to use the notion that most flights traverse a large geographic distance which will be discussed in greater detail next. This concept to conduct key distribution and revocation builds trust by getting the same key from multiple geographic locations.

GEOGRAPHY AIDED KEY DISTRIBUTION & VALIDATION

Aircraft operations can be used to aid in robust key distribution. Aircraft, by nature of their operations, traverse over large distances. As such it can gather keys from geographically distributed sources and these keys from these various sources can be used to verify authenticity. The cross check can just verifying that all keys received are the same. For example, the aircraft gathers and stores a key from its initial airport and perhaps any receivable source en route. As it approaches its destination airport or any location where it may need to authenticated, it gathers the key at that location. It then checks the currently received key against its stored keys. If the key is the same, then it is likely to be valid and can be used. Figure 6 sketches how the technique operates. This method can be applied to both GBAS and SBAS. It helps build trust in the received key and it can also be used to distribute new keys (thus revoking older ones). However, it does not solve the bandwidth issue.

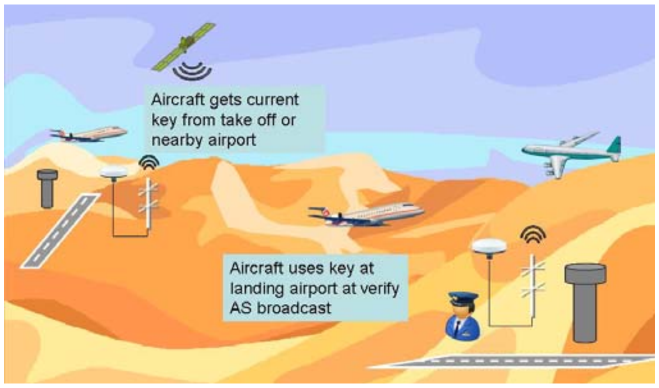
Aircraft gets current key from take off or nearby airport
Aircraft uses key at landing airport at verify AS broadcast

Figure 6. Aiding Key Distribution and Validation with Geographically Distributed Sources.

The proposed method has a couple of benefits. First, a spoofing any given user, requires spoofing the multiple geographically distributed locations that the user will gather keys from. This increases the cost and difficulty for the spoofer. An additional challenge for the spoofer is the fact that there will likely be other aircraft that have the true key since they come from locations that were not spoofed. These aircraft will be able to detect the presence of the spoofed key. This verification is applicable even if the spoofing target is at its initial airport. As a result, a spoofing must cover the entire coverage area of the augmentation system to have undetected spoofing.

For a SBAS system, such a technique makes sense as only a geostationary satellite can match its geographic range. However, as SBAS is used for almost all phases of aircraft operation, the method may not provide adequate confidence to keys for taxing and take off operations. This is because the aircraft will only have received keys in one region. This depends on the level of authentication security desired for such these operations as there are other factors (such as other aircraft which do have keys from multiple regions which can warn of local spoofing).

For GBAS, the strength or confidence that we can have increases as more airports adopt GBAS and the protocol. The more geographically separated locations that the key can be received, the more difficult it will be to spoof. However, there are two concerns. The first is a problem with initial implementation when there are only a few installed GBAS. Cross verification is not very effective if there are a few geographically distributed sources. The second is how to securely send the same keys to all GBAS stations which will be addressed next.

KEY DISTRIBUTION WITHIN AUGMENTATION SYSTEM

Another facet is distributing the proper keys to augmentation system segment responsible for broadcast. If the broadcast is distributed and not networked as may be the case with GBAS, one way would be to store the key and the mechanism to generate future keys within each ground station. Each GBAS station needs knows the current key as well as how to generate future keys and when they are to be applied for the lifetime of the installed station. Physical security such as having a tamper proof component or secure facility would be needed to prevent theft of the key. Key changes and revocation may be achieved with manual updates. For SBAS, the keys can resided within the master station as it is already responsible for generating and queuing messages.

CASE STUDIES

In the case studies, we examine how one may feasibly overlay an authentication capability on each augmentation system. As mentioned previously, key considerations to the design are bandwidth use, time to authenticate, key distribution/revocation and key length. The first two are related. Since the overhead needed to support authentication is the same regardless of the message size, more bandwidth available allows for more frequent authentication. In the studies, we will constrain the amount of bandwidth available as the designs should not significantly impact the distribution of integrity data. Given time to first fix for SBAS of two minutes, the time to authenticate should be no more than roughly 60 seconds. Even lower values is desirable as the typical time to alerts these SBAS and GBAS are even lower (2-10 seconds).

The last two considerations revolve around the key. The consideration is the need for key distribution and revocation. In these design studies, we will presume key distribution and verification can be achieved with minimal to no communications to a certificate authority. This may be achieved using means such as the proposed means to distribution using geographically diverse source to verify distributed keys. A mechanism also may also be needed for the revocation of broadcast keys. One means may be to provide keys (updated infrequently and perhaps offline) solely for the purpose of revoking a broadcast key. The second issue is key length. A key strengths equivalent to a symmetric key of 160 bits or more is likely necessary to support the system for roughly the next 20 years.

Again these designs are not meant to be suggestion but rather a first cut at seeing the reasonableness and cost of adding an authentication capability. The authentication is meant to be legacy compatible in the sense that current users can ignore the addition information.

SBAS

The addition of authentication on SBAS is challenging due to the bandwidth issues. First, much of its bandwidth is already used. While it may be possible to utilize the unused bits in some common messages, this amount of data is inadequate given the limited overall bandwidth. Likely, a design will have to be allocated some dedicated messages and bandwidth. The SBAS message format is seen in Figure 7. A maximum of 212 bits per message (with one message per second) is available. In the SBAS case studies, we limit the available bandwidth for authentication to roughly 10% of the total (6 messages per minute). Also, a key update (public key or base symmetric key) is performed at least every five minutes. These assumptions will help us quantify the trade offs in adding authentication.

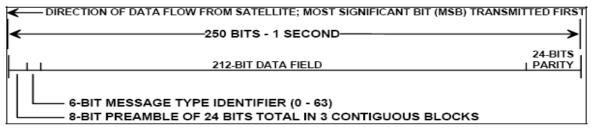
DIRECTION OF DATA FLOW FROM SATELLITE; MOST SIGNIFICANT BIT (MSB) TRANSMITTED FIRST
250 BITS - 1 SECOND
212-BIT DATA FIELD
24-BITS PARITY
6-BIT MESSAGE TYPE IDENTIFIER (0 - 63)
8-BIT PREAMBLE OF 24 BITS TOTAL IN 3 CONTIGUOUS BLOCKS

Figure 7. SBAS Message structure [11]

For a case study using public key, we examine implementing DSA or ECDSA. For either algorithm, domain parameters need to be shared between the user and the sender. These parameters can form part of the public key. These parameters require on the order of kilobits of data but they should not need to be changed. So ideally, they can be shared with the user *a priori* or through some other channel (i.e. regular update of electronic charts). This distribution needs be assured of validity.

However, a public key still needs to be distributed securely to the user. As seen in Table 5, the length of the public key for DSA is recommended to be 2048 (use up to 2030) or 3072 bits (use beyond 2030). For ECDSA, the key length is 224 and 256 bits for the two time frames, respectively. Given this, ECDSA is preferred. It will allow SBAS to send the public keys every few minutes using two messages and signatures using three messages. Given the bandwidth assumptions, this allows for a signature every 30 seconds with the public key broadcast every 5 minutes (10.7% or 32 out of 300 messages). If one minute time to authentication is acceptable, only 5.7% (17 out of 300 messages) of the SBAS bandwidth is required.

Time frame	DSA public key length (min, bits)	ECDSA public key length (min, bits)	Signature length (bits)
To 2030	2048	224	448 (2 at 224 bits)
Beyond 2030	3072	256	512 (2 at 256 bits)

Table 5. Recommended Key and Signature Length for DSA and ECDSA [12]

For the case study using symmetric key based authentication, we use the TESLA protocol. There are three things that need to be distributed to support TESLA: 1) Current base TESLA key, 2) MAC, 3) current TESLA key.

The key length should be sized such that the time to break the TESLA private key is less its exposure time. From Table 3, a key length of 160 bits or more seems acceptable. As a result, the key can fit in one message. The MAC itself can be the same length as the key though it can be truncated to half the key length. Truncating MACs to 106 bits, allows two MACs to be transmitted in one WAAS message. This helps maximize bandwidth usage. It can also mitigate the impact of message loss as each MAC is generated from a distinct subset of the messages. Figure 8 shows an implementation where a message can contain one or two MACs which are generated from the same MAC generation key, K'_{si}. If the message contains two MACs, one MAC is derived from the messages from the half of the time period and the other is derived from the messages from the later half.

An implementation may be achieved as follows. Send MAC message every m messages (for example, $m = 6$) with each message containing two MACs. This allows for better tolerance to lost messages. For $m = 6$, the user only needs 3 consecutive to authenticate rather than 6. The MAC update rate drives the bandwidth and the number of consecutive messages required for authentication. The current TESLA key every k MAC so that it is used to generate MACs for several message sets. This results in a time to authentication of (m+1)*k+1 seconds, provided the current TESLA key is sent immediately after the last MAC using the key. In Figure 8, the time to authentication is 2*(m+1)*k+1 as the key release is delayed by one until after the MACs generated using the next key are transmitted.

If the goal is roughly 20 second authentication, then k = 3 for m = 6. For 60 seconds, k = 9. Ten percent bandwidth usage is achievable with approximately 30 second time to authentication. This is done if MACs are sent every 15 messages and the current TESLA key is sent every two MAC message. It also means that time should be independently synchronized within 1 second since the key is sent in the message immediately after the MAC message. So there is only a one second gap between the release of the key and the final MAC generated using the key. Note that additional bandwidth is needed to send the base key.

The authentication may be strengthened by using old but active data (OBAD) or other previously verified information to aid in verification.

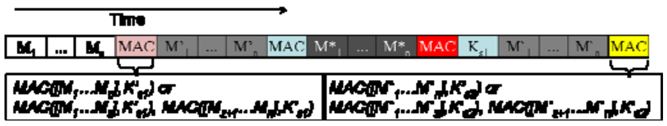

Figure 8. Example sequence for proposed scheme

From the study, it seems both ECDSA and TESLA offer viable with similar data rates as long as a certificate is not needed or requires minimal data. Some means of verifying the authenticity of the public key in ECDSA or the base TESLA key in TESLA is needed. The certificate is an important consideration if validating keys using geographically diverse sources is not adequate.

GBAS

Implementation of the authentication ideas on GBAS is simpler as it is less data constrained. The basic data components of the GBAS message are seen in Table 6. Two options are possible. First, unused bits from existing messages can be used. For example, GBAS Message Type 1 which provides corrections for up to 18 corrections (11 bytes each) has 7 byte unused. Roughly seven bytes every half second is not a lot of information but it is more than available in the SBAS case study (10% bandwidth equals 21.2 bits/second). As such the implementations discussed in SBAS can be used in GBAS provided the message structure is defined

The second option is to dedicate one message type and one message every *n* seconds for authentication. For ECDSA, a message can contain the signature, the public key and perhaps the certificate. Sending the message every five seconds uses ten percent of the bandwidth and allows for five second time to authentication. For TESLA, a message can easily contain multiple MACs as well as the base and a near current TESLA keys. Sending a message every five seconds will allow for a ten second time to authentication. The increased delay is because the key generating the MAC should be sent in a later message.

Message Block Segment	Bits	Bytes
Message Block Header	48	6
Message	Up to 1696	Up to 212
Message CRC	32	4
Total	1776 (max)	222 (max)

Table 6. Format of GBAS Message Block [2]

CONCLUSIONS

This paper studies the feasibility and means by which authentication can be overlaid upon the existing SBAS and GBAS designs. It considers how to achieve the authentication that is compatible with the current augmentation system and its users. It also considers how to perform the security necessary to support authentication within the current NAS framework. One important issue is secure key distribution and the paper presents some options designed to be reasonable for aviation infrastructure and operations. One means is a key distribution protocol that utilizes the operation of the aircraft and air traffic to aid in key verification. This provides to distribute keys and provide some ability to validate them without significant additions to the NAS. Another issue is bandwidth. The paper presents ways of modifying protocols such as TESLA to reduce bandwidth use while maintaining an acceptable level of security.

The paper uses the current L1 SBAS and GBAS as case studies. The paper presents reasonable method to provide authentication on the current SBAS using about ten percent of bandwidth. The method is compatible to current SBAS user equipment in that they will not be adversely affected. GBAS can employ similar means. As it has greater data bandwidth, a more critical issue for GBAS is key distribution to the ground stations.

DISCLAIMERS

ACKNOWLEDGMENTS

The authors would like to thank Leo Eldredge and Mitch Narins of the FAA Navigation Services Directorate for supporting this work.

REFERENCES

[1] Murphy, T., Harris, M., Burns, J., “Modifications to GBAS for VDB Authentication”, Navigation Systems Panel, CAT III Subgroup Meeting, July 8-10, 2008.

[2] RTCA SC-159, “GNSS Based Precision Approach Local Area Augmentation System (LAAS) – Signal-in-Space Interface Control Document (ICD)” ‘DO-246D December 2008.

[3] Schneier, Bruce, “Applied cryptography: protocols, algorithms, and source code in C ,” 2nd ed., Wiley, New York, 1996.

[4] Katz, Jonathan, "Introduction to Modern Cryptography: Principles and Protocols,” Chapman & Hall/CRC, Boca Raton, FL, 2006.

[5] National Security Agency, Central Security Service, “The Case for Elliptic Curve Cryptography” http://www.nsa.gov/business/programs/elliptic_curve.shtml, January 2009

[6] Perrig, A. Canetti, R., Tygar, J.D., and Song, D., “The TESLA Broadcast Authentication Protocol,” CryptoBytes, 5:2, Summer/Fall 2002, pp. 2-13

[7] Wullems, C., Pozzobon, O., Kubik, K., “Signal Authentication and Integrity Schemes for Next Generation Global Navigation Satellite Systems,” Proceedings of the European Navigation Conference GNSS, Munich, July 2005

[8] Kuhn, Markus G., “An Asymmetric Security Mechanism for Navigation Signals”, 6th Information Hiding Workshop, 23-25 May 2004, Toronto, Canada, Proceedings, LNCS 3200, pp. 239–252, Springer-Verlag.

[9] Qiu, Di, Lo, Sherman, Enge, Per, “Geoencryption using Loran”, Proceedings of the Institute of Navigation National Technical Meeting, San Diego, CA, January 2007

[10] Bellare, M., Canetti, R., and Krawczyk, H. “Keying Hash Functions for Message Authentication,” Crypto 96 Proceedings, Lecture Notes in Computer Science, Vol. 1109, Spring Verlag, 1996 citation

[11] RTCA SC-159, Minimum Operational Performance Standard for Global Positioning System/Wide Area Augmentation System Airborne Equipment, RTCA/DO-229D, December 2006.

[12] National Institute of Standards and Technology, “Recommendation for Key Management – Part 1L General (Revised),” Special Publication 800-57, March 2007
.