Authentication of the medium frequency R-Mode navigation message

Stefan Gewies
Department of Nautical Systems
German Aerospace Center
Neustrelitz, Germany
Stefan.Gewies@dlr.de

Thomas Strang
Department of Institute Project Management and Administration
German Aerospace Center
Oberpfaffenhofen, Germany
Thomas.Strang@dlr.de

***Abstract*— Medium frequency R-Mode is a maritime terrestrial navigation system which supports the mariner with alternative positioning and timing in the event of Global Navigation Satellite Systems (GNSS) unavailability or performance reduction. The paper presents a proposal to increase the security of the R-Mode service by introducing a cryptographic signature and validation scheme for the R-Mode navigation messages.**

Keywords — R-Mode, navigation message, authentication, TESLA, Secure APNT

I. INTRODUCTION

Ranging mode, known as R-Mode, is an alternative maritime navigation system which utilizes existing maritime radio infrastructure for the transmission of synchronized ranging signals [1] [2]. Possible infrastructures are maritime radio beacons [3] [4] [5], automatic identification system (AIS) base stations [6] [7] and very high frequency data exchange system (VDES) base stations [8] [9]. R-Mode is designed as a maritime backup system for Global Navigation Satellite Systems (GNSS), which are today the backbone of maritime navigation at sea, in coastal areas and in ports. Man-made GNSS interferences through jamming and spoofing are a serious threat for shipping and, indirectly, for the environment and economy of countries such as Germany which depend on smoothly functioning sea trade. Furthermore, it affects the operation of navy, federal police and search and rescue (SAR) vessels.

This paper focuses on the maritime radio-beacon component of the R-Mode system which operates at medium frequencies (MF) at around 300 kHz. MF R-Mode is a system component under development. The time signature of the R-Mode ranging signal is well defined [10] and implemented in three testbeds in Canada, Republic of Korea and the Baltic Sea. Ongoing validation measurements show that MF R-Mode has the potential to support coastal navigation with 95% positioning accuracy of about 20 m in the daytime and 60 m at nighttime [11].

Besides the time signature, the R-Mode signal also carries navigation information to the user which is minimum-shift keying (MSK) modulated [12]. It is proposed that the data are broadcast with 100 bps in Europe and encoded with the RTCM 10402.3 [13] and ITU M.823 [14] standards. For compatibility with the legacy radio-beacon service, a GNSS support with code differential GNSS (DGNSS) corrections, both (DGNSS corrections and R-Mode) have to share the available bandwidth which limits the available data rate for the R-Mode navigation information to about 50 bps.

The aforementioned man-made interferences (jamming and spoofing) can be observed these days to GNSS in the Baltic Sea area (Fig. 1). Because R-Mode is also a radio navigation system, spoofing has to be properly addressed in R-Mode as well. More precisely, any R-Mode user must be able to verify that relevant R-Mode messages are not modified and are from the original authorized sender. Both can be addressed with cryptographic methods to ensure the security of the R-Mode system on R-Mode navigation message level.

A straightforward approach would be to add a cryptographic signature to each MF R-Mode message using a public-private-key-based authentication mechanism. However, this approach is not feasible for a variety of reasons, including computationally expensive asymmetric cryptographic algorithms, complex key management and particularly large message sizes. Instead, we propose to use an adaptation of the TESLA broadcast authentication protocol [15], which has been shown to be a feasible authentication scheme for broadcast communications in other modes of transport, namely aviation [16]. The basic idea of TESLA is to use a key to cryptographically sign a series of messages, which is then published after a certain time interval. This key is derived from a cryptographically linked key chain and only known to the sender before publishing. The sender calculates a Hash-based Message Authentication Code (HMAC) for every message using symmetric functions applying the key of the current interval, and adds that HMAC (but not the current key) to each message. With a certain defined delay, the user receives the required key and can then verify the integrity of the previously received message using its attached HMAC. Hence, combining fast symmetric cryptographic measures,

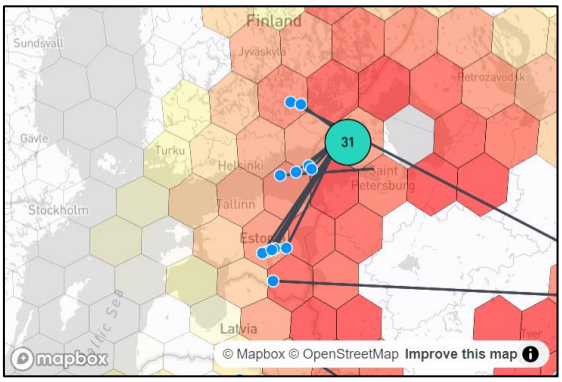

Figure 1: Reduced Global Positioning System (GPS) performance in the Gulf of Finland area for August 20th 2024 as reported by airplanes. 31 airplanes were affected by spoofing identified by the algorithms of SkAI Data Services and the Zurich University of Applied Sciences [20].

short HMACs and properly derived key chains with a delay in the publication of the key creates a signature-like scheme.

The paper elaborates on the details of the approach, some of the essential parameters such as suitable key sizes, delay between key publication and other aspects, which will ensure the general feasibility of the approach as well as the applicability to existing components including backward compatibility.

II. MF R-MODE

The MF R-Mode system utilizes the radio-beacon DGNSS stations which are typically distributed along the coastline of areas with high traffic density. The distances between the sites are usually such that the coastline in these areas is often covered twice or more. Especially water areas partially enclosed by land or with islands or larger bays with dimensions of a few 100 km or less are usually well suited for MF R-Mode, because ships can receive the signals of three or more radio beacons. So instead of the discontinuation of the radio-beacon DGNSS service as decided by some countries (e.g. USA, United Kingdom, Australia and Japan), their co-usage for MF R-Mode provides a valuable Alternative Positioning Navigation and Time (APNT) service to the maritime users.

R-Mode is a time of arrival (TOA)-based positioning and timing system. Each transmitting station is synchronized to the R-Mode System Time (RMST) and the RMST itself is traceable to UTC. All R-Mode signals are transmitted with respect to RMST. Any signal delays will be measured and reported as error with respect to the RMST. A multilateration algorithm is used to determine the position and time when at least three R-Mode signals from different directions can be received. This can be any combination from MF and VDES R-Mode signals. The MF R-Mode ranging and positioning accuracy depends particularly on the geometry of the transmitting stations, the distance between the receiver and transmitting stations, the electrical parameters of the Earth's surface along the propagation path, the radio noise in the maritime MF frequency band and the occurrence of multipath of the space wave at night.

The transmitted MF R-Mode signal extends the transmitted minimum-shift keying (MSK) modulated signal of the maritime radio beacons. Two aiding carriers at 225 Hz below and above the carrier frequency of the radio beacon are added to their continuous transmission. The R-Mode signal components are aligned to the RMST by the signal definition of ascending zero crossing of both aiding carriers and MSK bit transition at the beginning of each second. Any deviation from the specification will be provided by the R-Mode navigation message. At the receiver side, the phase of the aiding carriers will be used to perform pseudo range estimation [3].

R-Mode extends on RTCM 10402.3, a standard to provide differential correction data for GNSS receivers via radio beacons which is commonly referred to as RTCM v.2.3 [13]. It adds an additional message type to the existing ones. In order to not interfere with the existing GNSS supplementary service, the R-Mode message was designed as follows.

RTCM 10402.3 is based on a set of fixed-length 30-bit "words" which are combined into longer messages known as "frames". All words end with a 6-bit "parity" code using the same algorithm as the original GPS signals, based on Hamming codes. This leaves 24 bits available for data per word. The format was deliberately modelled on that of the GPS messages, in order to maintain familiarity. Data within the 24-bit payload are extracted into individual data and then encoded for local transmission as strings of 6-bits of data with a leading 1 start bit and trailing 0 stop bit to form a single 8-bit value suitable for use on ASCII-based serial links and similar. The parity bits allow for some level of forward error detection and correction in case of single bit errors.

All of the RTCM 10402.3 frames start with a standard two-word header. The first 24+6-bit word starts with a magic number, a fixed 8 bit "preamble". The next 6 bits encode the message type, 0 to 63. This is followed by a 10-bit transmitting station ID. The second RTCM header word begins with a 13-bit version of the modified z-count, which is the unit of time in GPS, followed by a 3-bit frame sequence number, a 5-bit length that counts the total number of 24+6-bit words in the frame following the header (0..31), and a 3-bit "station health" code. As each RTCM 10402.3 frame can consist of up to 33 words including 2 header words, the maximum length of one frame is 33x 24+6 bits = 990 bits.

For the purpose of R-Mode, a dynamic RTCM 10402.3 message with the ID 55 was defined [12]. For compatibility reasons, the proposed R-Mode navigation message uses the same two RTCM header words and parity bits, the shortest RTCM message available. The R-Mode transmitting station then adds another 24+6-bit header with all the important R-Mode status information that has to be provided most frequently, i.e. with an update rate of approximately once per 5 seconds. This encodes submessage 0, which provides only status information and timing of the transmission.

Submessage ID	0 = no additional information 1 = RMST week, signal delays and offset (3 words) 2 = Static navigation data (3 words) 3 = RMST to UTC conversion (5 words) 4 = Free running clock offset (2 words) 5 = Differential R-Mode corrections (2 words) 6 = Static navigation data for Differential R-Mode station (3 words) 7 = *not used*

Figure 2: R-Mode submessage types as defined in the R-Mode navigation message proposal [12]

All other information supporting a navigation method alternative to DGNSS is encoded in six predefined R-Mode submessages (1 to 6 in Fig. 2), which extend the R-Mode message by two to five words, each with a length of 30 bits, using the same encoding scheme and which will be transmitted with the target rates as indicated in Table 1.

The update rates were defined based on the following assumption: The R-Mode message has to be integrated into the DGNSS data stream. Due to the length of certain (non-R-Mode) DGNSS messages, the next possible transmission of an R-Mode navigation message has to wait for up to a few seconds. To make sure that the R-Mode receiver obtains at least one R-Mode status information within 10 s, the transmission of R-Mode status is desirable each 5 s. In the case of a cold start, the receiver should obtain all the information necessary to perform R-Mode-based positioning within one minute. Interoperability with other navigation systems should be possible after five minutes. To not interrupt the DGNSS correction stream for too long, the maximum length of message 55 was therefore restricted to a total length of 240 bits, including all RTCM 10402.3 and R-Mode headers (8 words). This implies a transmission time of up to 2.4 s for 100

bit/s radio-beacon transmission bit rate or, in other words, the DGNSS correction data stream will be interrupted for up to 2.4 s for each R-Mode message.

TABLE 1: R-MODE SUBMESSAGE TARGET RATES AND MESSAGE SIZES AS DEFINED IN THE R-MODE MESSAGE PROPOSAL [12]

R-Mode Submessage	**Approx. Rate**	**Gross-Size**	**DGNSS Interr.**
0 – status	1 per 5 sec	90 bit	0.9 s
1 – clock correction	1 per 1 min	180 bit	1.8 s
2 – static nav data	1 per 1 min	180 bit	1.8 s
3 – time relation	1 per 5 min	240 bit	2.4 s
4 – clock offset	1 per 1 min	150 bit	1.5 s
5 – differential correction (opt.)	1 per 1 min	150 bit	1.5 s
6 – diff. static nav data (opt.)	1 per 5 min	180 bit	1.8 s

This proposed extension of radio-beacon transmission was sent to the International Association of Marine Aids to Navigation and Lighthouse Authorities (IALA) for MF R-Mode standardization [12].

III. Potential attacks to R-Mode

Without additional protection, R-Mode is prone to a similar range of attack vectors as GNSS. For instance, relatively simple yet effective jamming attacks trying to achieve R-Mode denial of service are unlikely to remain undetected, as any strong source of noise can be relatively easily detected by continuous monitoring of the signal-to-noise ratio of one or several signal components even at the receiver side. In addition, the far-field monitor as part of the R-Mode infrastructure checks the quality of transmitted R-Mode signals and can flag abnormal signals identified. Even the source of such interferences is not hard to identify, as any transmitting antennas in the MF frequency range are typically several tens of meters in size and the transmitting power is in the order of 100 W.

An attack on one or both aiding carriers of the R-Mode signal can take place with signals which contain components of one frequency or both frequencies. The superposition of the original R-Mode signal with the interference signal for the same frequency will generate a sum signal with the same frequency, but different phase and amplitude. A varying amplitude is expected depending on whether the vessel is moving or the source of interference is moving. Accurate tracking of the phase and signal amplitude as well as comparison of the amplitudes of all three R-Mode signal components can be used to detect the interference.

Whereas jamming attacks resulting in a service non-availability are at least relatively easy to detect, all other attacks – in particular spoofing of the R-Mode navigation message – make it very obvious that the use of non-authenticated data is very prone to man-made interferences and justify the use of a feasible authentication scheme. The potential impact of compromised data on the receiver signal processing, pseudo ranges and position estimation can be manifold and can lead to e.g. a vessel colliding with a sand bank. The authentication of data is the gold standard for detecting such attacks before they result in anything critical.

IV. TESLA Broadcast Authentication Scheme

The TESLA (Timed Efficient Stream Loss-tolerant Authentication) Broadcast Authentication Protocol [15] was introduced in 2002 by Perrig at al. as an efficient protocol with low communication and computation overhead, which scales to large numbers of receivers, and tolerates packet loss. TESLA is based on loose time synchronization between the sender and the receivers, using purely symmetric cryptographic functions, but achieving asymmetric properties. The main idea of TESLA is the delayed disclosure of a key k known only to itself that is used to calculate a key-dependent message authentication code (MAC) from a hash of the data to authenticate, which the sender e.g. attaches to the packet. The receiver buffers the received packet without being able to authenticate it right away. A short while later, the sender discloses k which only then enables the receiver to authenticate the packet. Consequently, a single MAC per packet suffices to provide broadcast authentication, given that the receiver has (roughly) synchronized its clock with the sender ahead of time.

The keys k_i to be used in each interval i of uniform duration T are taken from a one-way chain of length l generated at the sender with a suitable one-way hash function F by applying l times the function F from a random start value k_l, using the output of F as input for the next round down to k_0. The keys from the chain are used in reverse order of generation. The sequence of k_i is kept secret at the sender's side up until publication after usage in one of the previous intervals in reverse order as keys to calculate a key-dependent MAC of the hash of the packet. A nice feature of this approach is that from each newly disclosed k_i any receiver can trace back to any previously disclosed k_{i-x} down to k_0. This enables verification of whether the packet has been received from the same sender not only for the previous interval (x=1), but also (e.g. in case of packet loss during transmission) for any interval since the start of the chain (i=0).

The sender uses a defined disclosure time for the one-way chain values, usually expressed as a factor d of the interval T with d being called the disclosure delay. The sender publishes the value after the disclosure time, e.g. along with any of the packets. Each receiver that receives a packet containing a newly disclosed key can check whether this key is authentic by re-calculating up to l times F from the received newly disclosed key. If it matches any of the previously disclosed keys, it can be considered as having been issued by the same source as the previous one, i.e. it belongs to the same chain owner (which might or might not be the assumed identity at this stage). Note that Perrig et al. [15] also introduced a second key-derivation function F', which is used to derive a key k_i' from each chain element k_i following a best practice of not "*using the same key multiple times in different cryptographic operations [as this] is ill-advised - it may lead to cryptographic weaknesses*". As a result, not the keys k_i of the chain itself are used to calculate the MAC at both sides (sender and receiver), but the derived keys k_i' applying the same public functions F and F'.

Any receiver can check whether each packet received during interval i was "signed" with the key k_i, which is only disclosed in the next (d=1) or any later (d>1) interval. Only if this is the case, can the packet be considered as having been sent by the same chain owner who published the key. Note that the receiver can immediately use the content of any packet (as long as it is not encrypted in addition) after reception, but should consider the risk of an ongoing spoofing attack until the MAC can be validated in a later interval. In particular, the receiver has to validate the chain owner as being authentic, i.e.

the authorized identity, in addition. This can be done with an additional asymmetric signature of any chain element in any arbitrary way. How any necessary certificates of such an asymmetric scheme are distributed (e.g. using a PKI) is outside the scope of the TESLA protocol. Only after this secondary verification step can the receiver be sure that not only the MAC-signed packets were signed by the same, but also the valid chain owner, eventually rendering the packet to be authentic. Thus, to perform the TESLA authentication scheme, the received messages have to be stored for delayed validation, which also requires sufficient storage capacity at the receiver side. Note that the scheme is tolerant of the loss of single messages as from any next correctly received message, e.g. containing a key of the chain, the full chain down to k_0 can be reconstructed by the receiver. And with any next receival of the asymmetric signature of k_0, the chain owner can be authenticated again.

The German Federal Office of Information Security (BSI) provides an assessment of the security of state-of-the-art cryptographic mechanisms in the light of a long-term orientation for their use [17]. This include requirements for all cryptographic primitives and key sizes used in TESLA. It is strongly advisable to follow these recommendations as they are based on the latest identified security weaknesses of the art. The current minimum security level to be achieved by any cryptographic mechanism shall be at least 120 bits, meaning that there are costs associated with each possible attack against the mechanism that breaks the mechanism's security objective with a high probability of success equivalent to 2^n calculations of the encryption function of an efficient block cipher. This renders into a minimum key size of at least 120 bits for an ideal symmetric MAC algorithm, as well as a minimum key size of at least 240 bits for calculating the signature in an asymmetric scheme such as Elliptic Curve Digital Signature Algorithm (ECDSA). The latter would result in a signature length of two times the key length, i.e. 480 bits. Other security parameters beside the key lengths in the TESLA scheme include the length of the digest output of the MAC algorithm (so-called MAC tag length). According to the BSI [17], "*Ideally, a MAC should in practice be indistinguishable for an attacker from a random function with a corresponding digest length. As long as this criterion is met, the attacker is left with the option of generating fake messages by guessing, with a per-attempt probability of success of 2^{-n} when n is the tag length. In many applications, n=96 can be considered acceptable in such a situation.*" However, as the MAC tag is calculated on a hash of a message of arbitrary length (in turn then called HMAC), the hash function itself should be cryptographically strong, such as SHA-256, efficiently and securely calculating a 256-bit "fingerprint" of any input of arbitrary length with very low collision probability.

V. TESLA Authentication for R-Mode

Following the recommendations of the BSI, we propose to use SHA-256 as an efficient yet secure state of the art hash function to generate a key chain at the sender out of an arbitrary random key k_l. SHA-256 works on input block sizes of 512 bits. Any input shorter than 512 bits is padded internally until it reaches a multiple of 512 bits. As for the elements of the key chain, we propose to go with a key size of 128 bits to maintain the minimum security of 120 bits or more as advised by the BSI. That means for every $0 \leq i < l$, the sender calculates $k_i = F(k_{i+1})$ using $F(x) = \text{TRUNC}_{128}(\text{SHA-256}(x))$. Here, $\text{TRUNC}_y(z)$ denotes the last y bits of the input z. Each resulting k_i with a length of 128 bits shall be assigned to the time interval with the respective index i to be used for calculating the HMAC before publishing in one of the later intervals using the format illustrated in Fig. 3. Time-synchronization required by TESLA is already part of the underlying R-Mode core approach.

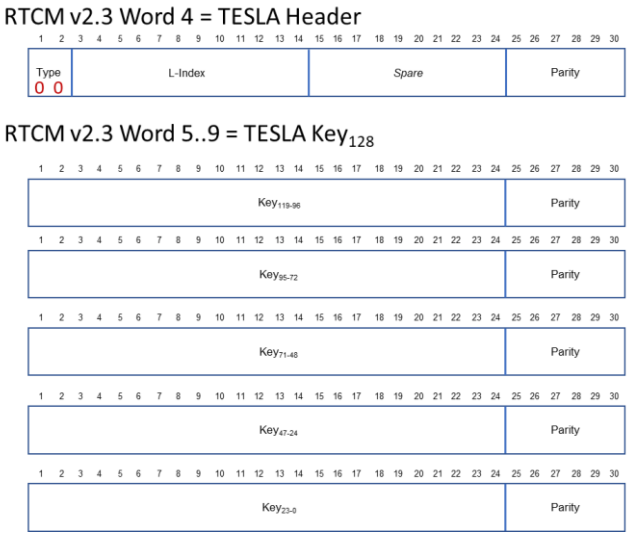

Figure 3: R-Mode TESLA key publication message format

To enable any R-Mode user to distinguish authentic, non-modified messages (navigation and/or authentication) from intentionally or unintentionally modified messages, including those from misbehaving senders, we propose to adopt the TESLA scheme as already proposed by Lázaro et al. [18] for VDES R-Mode. However, we do deviate from Lázaro et al. [18] in several aspects to further strengthen the approach from the security perspective while addressing the lower available data rate of MF R-Mode at the same time. Therefore, instead of adding an individual 15-20-bit MAC tag to each single R-Mode message as proposed by Lázaro et al. [18], we follow the recommendations of the BSI, as described above, by calculating an HMAC of one or more R-Mode messages with a tag length of 96 bits, which shall be transmitted as an individual TESLA message after the R-Mode message(s). For this purpose, we propose to use a spare submessage type 7 of the R-Mode header (see Fig. 2) to introduce a new R-Mode TESLA message following the general RTCM 10402.3 message format. This TESLA message starts with a header indicating one out of four different TESLA message types, here "01" being the signature message (see Fig. 4).

To account for the flexibility of indicating which of the last R-Mode navigation message(s) shall be protected with the current signature and are thus included in the calculation of the MAC tag, a respective bit field ("inclusion") is provided. Note that each TESLA message is preceded by the R-Mode header and, thus, the latest R-Mode status information (submessage type 0 if sent individually) is always included, updating any status provided with any other R-Mode navigation message of subtypes 0..7 provided before. Thus, the latest status as represented in the R-Mode header of the actual signature message is always included and the inclusion field consists of 7 bits to be used to mark the last instance of any other R-Mode submessage 1..7 to be included or not.

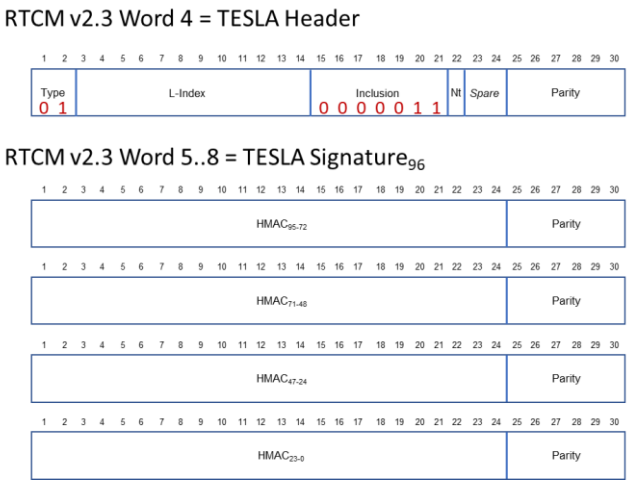

Figure 4: R-Mode TESLA signature HMAC message format

Following the good practice of not using the same key multiple times in two independent cryptographic operations as already foreseen in the original TESLA proposal [15], we do not directly use the output of the hash function F to calculate the MAC tag, but a key k'_i derived from k_i on both sides (sender and receiver) using $F'(k_i) = k'_i = \text{SHA-256}(k_i)$. The HMAC itself is the last 96 bits of the operation

$$\text{HMAC}_{96} = \text{TRUNC}_{96}(\ \text{SHA-256}\ (\ k'_i \parallel \text{for all } s{=}0..7\text{: submsg}(s) * \text{inclusion}(s)\ ))$$

The sender (respectively their authority) can define which R-Mode messages shall be signed individually or collectively and how often the signatures shall be distributed. Future versions of the proposed protocol may use a different key derivation function for $F'(k_i)$ (and maybe even $F(k_i)$) such as HKDF [19]; however, as there appears to be no security advantage for the time being, we build upon the wide support of SHA-256 on any platform.

From a security perspective, it would be ideal to send a signature after every single R-Mode navigation message, minimizing the need to correctly receive more than one message to perform an HMAC validation. However, this also consumes the highest share of the available data rate. To the other extreme, the sender could generate HMAC tags of the latest instance of all different R-Mode navigation message types or even skip some instances before transmitting a new HMAC tag. In particular the latter would significantly weaken the security of the authentication approach, but minimizes the share of the data rate required by the protocol.

The signature message format accounts for additional header information, such as whether the net data (24 bits) or the gross data (30 bits) of all RTCM words of the protected messages are taken as input for the HMAC calculation. The net version would also allow authenticity verification of re-constructed messages after some kind of forward error correction applied to a limited amount of bit errors using the parity bits of the original RTCM stream, whereas the gross version is a little less complex in handling.

To finally verify that the owner of the chain is authentic, we propose to use the R-Mode TESLA message to publish an ECDSA signature of k_0 of the current chain using a private ECDSA key of 256 bits once in a while. k_0 can be derived by the receiver from any correctly received k_i. How private/public ECDSA keys are generated, distributed and updated (usually involving some Public Key Infrastructure) is outside the scope of this paper and does not affect the TESLA scheme. Every ECDSA signature is twice as long as the key used, i.e. 512 bits following our proposal, and consists of two parts called r-value and s-value. We propose to send both parts with individual R-Mode TESLA messages according to Fig. 5.

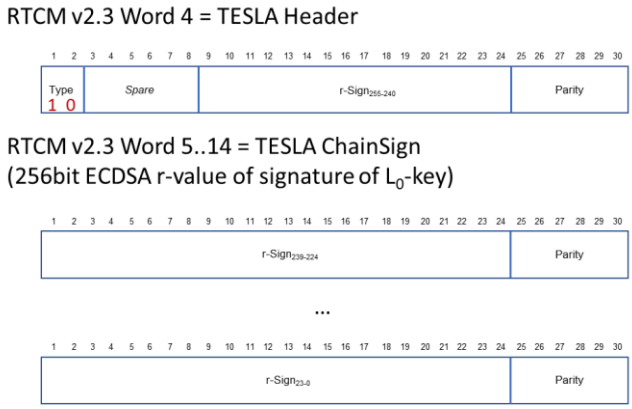

Figure 5: R-Mode TESLA ECDSA signature r-value with type "10" (l-value analogue with type "11", not shown)

R-Mode TESLA messages containing an ECDSA signature are sized as 2x 14 words and thus much longer than any of the other R-Mode TESLA messages. However, they need to be known to the receiver only once to validate an entire key chain. Thus, the main driver of the required repetition rate is an acceptable time to a first full authentication after starting a receiver, receiving a new radio beacon for the first time or starting a new key chain.

VI. Possible Attacks to TESLA-protected R-Mode

A deliberate choice of adequate cryptographic primitives and key sizes have been chosen to withstand cyber-attacks of the art. For the generation and use of the key chain central to TESLA, this includes, in particular, a sufficiently strong hash function (SHA-256), key sizes greater than or equal to the current recommended minimum security level of 120 bits, and a MAC tag length of $\geq$ 96 bits. A sufficiently strong asymmetric mechanism for chain-ownership validation has been proposed. However, any cryptographic approach is as strong as the primitives behind it. Also, the choice of transmission rates of the different R-Mode TESLA messages as well as the referred R-Mode navigation messages contribute to the level of achievable security.

One possible attack on a TESLA-protected R-Mode could aim at entrapping the receiver to accept a key from an attacker as being part of the key-chain of a valid transmitter, published during the interval $i+d$, e.g. using a fake transmitter with dominant power mimicking most of the data of the original transmitter. Any receiver can trace back whether the newly arrived (false) key can be used to calculate any of the previous chain elements down to k_0. If that is not the case, the newly arrived key must not be accepted, the attack is detected and any data received from the associated transmitter since interval i must be deemed invalid. The only way to evade detection would be if the attacker finds a colliding input for the underlying hash function used during key chain generation, which is extremely unlikely due to the design of SHA-256.

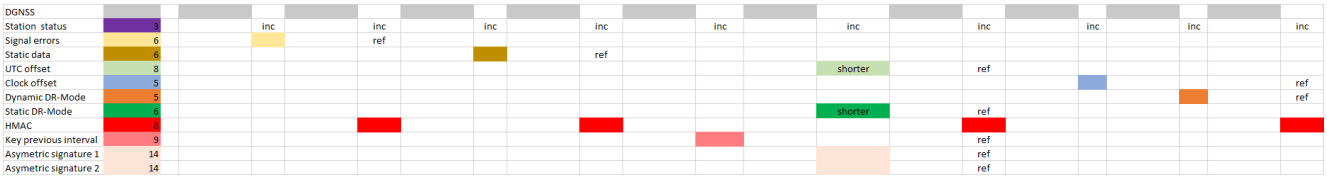

Figure 6: RTCM stream outline containing R-Mode navigation and TESLA messages

Another possible attack could be in finding a MAC which satisfies $F'=H(k'_i||M'_i)$ given a "plausible" M'_i which is content-wise similar to an original M_i to be transmitted in interval i, even without the need to know k'_i. Again, this is very unlikely to be possible, given the construction of F'. For non-plausible M'_i it is very likely that the attack can be easily detected (e.g. big "jumps" in the position calculation).

Replay attacks, i.e. re-transmitting stored received original TESLA messages, are non-harmful, given that each message already contains a timestamp of the message as part of the RTCM 10402.3 header. Of course, any receiver of RTCM messages has to check whether any received message is related to the current time, including any TESLA message – and if not, discard the message. A little trickier is what could be called a re-located re-usage attack. Here, a malicious user could receive and store a valid key from an original transmitter after key release, and use that key in a different geographical area, i.e. outside the reception area of the original user, to mimic authentic messages originating from the original user in that other area, but with malicious content. The risk of immediate detection at the receiver without further measures is limited in regions without centralized communication. This urges the need to provide (and co-sign) not only k_0 by the chain owner, but also the point in time of starting the chain and the duration of the interval, so that any receiver can calculate whether a key received at a certain time is really due in the current interval. This is again considered to be a mechanism aligned to the private/public ECDSA key distribution and thus out of scope for this paper.

VII. Operational Aspects

One important boundary condition for MF R-Mode is that the existing legacy service, to provide code differential GNSS correction and integrity information, should continue to be available without restrictions. It is expected that R-Mode can use up to 50% of the available data channel. This value can be different depending on the data rate of the radio beacon (100 bps or 200 bps), the number of supported GNSS constellations and the used message types. If the R-Mode navigation message is implemented as given in Table 1, it will cause a utilization of around 30% of a 100 bps data channel, which is typical for a radio beacon in Europe.

One possible schedule for R-Mode with TESLA extension could be the following. It assumes a DGNSS data channel utilization of 60%. A minute of RTCM 10402.3 message stream containing R-Mode navigation and R-Mode TESLA messages may look similar to Fig. 6 if, for example, HMACs are sent roughly every 15s, keys published roughly every minute and ECDSA signatures repeated roughly every 4 minutes. Note that there is no need for submessage 0 R-Mode messages, as they are subsumed by all R-Mode navigation and TESLA messages.

This scheme consumes a further 23% of the available bandwidth for the TESLA extension. However, as all TESLA messages include the R-Mode header, there is significantly less need for additional submessage 0. The example in Fig. 6 illustrates that 10 out of 12 submessage 0 can already be omitted without losing any update, resulting in a combined data rate share of 38% for TESLA-protected R-Mode, i.e. well below the 50% threshold.

Backward compatibility to R-Mode as well as RTCM 10402.3 is fully given; however, as every HMAC is also a checksum of the data hashed, parity bits could be traded for a more efficient encoding of HMAC, keys and signatures, but losing backward compatibility.

VIII. Conclusion

In this paper, we used a spare submessage type in the MF R-Mode specification to introduce a new R-Mode TESLA message to sign one or more previous R-Mode message(s) with an HMAC and a key derived from a not-yet published element of a TESLA key chain (step 1), to publish the element of the chain used to calculate the HMACs in the previous interval (step 2) and to validate the chain owner with an asymmetric signature (step 3).

All BSI criteria on secure algorithm selection, key sizes and MAC tag sizes are met. Single R-Mode messages may be lost without affecting the ranging or authentication of future successful transmissions.

The approach is currently at a conceptual level; reference implementation and tests are yet to come. A future extension of the approach is to authenticate not only R-Mode navigation messages, but any kind of RTCM 10402.3 messages, in particular also DGNSS correction messages.

References

[1] G. Johnson, P. Swaszek, J. Alberding, M. Hoppe and J.-H. Oltmann, "The Feasibility of R-Mode to Meet Resilient PNT Requirements for e-Navigation," *Proceedings of the 27th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2014)*, pp. 3076 - 3100, 8 - 12 September 2014.

[2] S. Gewies, A. Dammann, R. Ziebold, J. Bäckstedt, K. Bronk, B. Wereszko, C. Rieck, P. Gustafson, C. G. Eliassen, M. Hoppe and W. Tycholiz, "R-Mode Testbed in the Baltic Sea," in *IALA conference*, Incheon, 2018.

[3] L. Grundhöfer, F. G. Rizzi, S. Gewies, M. Hoppe, J. Bäckstedt, M. Dziewicki and G. D. Galdo, "Positioning with medium frequency R-Mode," *Navigation, Journal of the Institute of Navigation,* vol. 68, no. 4, pp. 829-841, 6 December 2021.

[4] S. Lee, Y. Han, S. H. Park, K. Seo and T. H. Fang, "Feasibility Analysis of R-Mode in R.O.K. from MF Beacon Station Deployment," in *IALA Conference*, Incheon, 2017.

[5] G. Johnson, K. Dykstra, S. Ordell and P. Swaszek, "R-Mode Positioning System Demonstration," *Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, pp. 839 - 855, 21 - 25 September 2020.

[6] Q. Hu, Y. Jiang, J. Zhang, X. Sun and S. Zhang, "Development of an Automatic Identification System Autonomous Positioning System," *Sensors*, vol. 15, pp. 28574-28591, 2015.

[7] K. Bronk, P. Koncicki, A. Lipka, R. Niski and B. Wereszko, "Concept, signal design, and measurement studies of the R-mode Baltic system," *Journal of the Institute of Navigation*, vol. 68, no. 3, pp. 465-483, 2021.

[8] K. Bronk, M. Januszewska, P. Koncicki, A. Lipka, R. Niski and B. Wereszko, "Ranging and Positioning Accuracy for Selected Correlators under VHF Maritime Propagation Conditions," *Journal of Telecommunications and Information Technology*, no. 3, pp. 3-15, 2022.

[9] M. Wirsing, A. Dammann and R. Raulefs, "Direct Position Estimation for VDES R-Mode," *2023 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pp. 724-728, 2023.

[10] P. F. Swaszek, G. W. Johnson, J. Alberding, M. Hoppe and J. Oltmann, "Analysis of MF-DGNSS Modifications for Improved Ranging," in *ENC*, Rotterdam, 2014.

[11] F. G. Rizzi, L. Grundhöfer, S. Gewies and T. Ehlers, "Performance Assessment of the Medium Frequency R-Mode Baltic Testbed at Sea near Rostock," *Applied Sciences*, vol. 13, p. 1872, 2023.

[12] IALA, "Guideline on Medium Frequency R-Mode signal structure and navigation message (draft)," IALA, Saint-Germain-en-Laye, 2024, unpublished.

[13] RTCM, “RTCM 10402.3 Recommended Standards for Differential GNSS (Global Navigation Satellite Systems) Service, Version 2.3,” 20 August 2001.

[14] ITU-R, “Recommendation ITU-R M.823-3 - Technical characteristics of differential transmissions for global navigation,” 2006.

[15] A. Perrig, R. Canetti, J. D. Tygar and D. Song, "The TESLA Broadcast Authentication Protocol," *RSA Cryptobytes*, vol. 5, no. Summer/Fall 2002, p. 2–13, 2002.

[16] N. Mäurer, T. Gräupl, M. A. Bellido-Manganell, D. M. Mielke, A. Filip-Dhaubhadel, O. Heirich, D. Gerbeth, M. Felux, L. M. Schalk, D. Becker, N. Schneckenburger and M. Schnell, "Flight Trial Demonstration of Secure GBAS via the L-band Digital Aeronautical Communications System (LDACS)," *IEEE Aerospace and Electronic Systems Magazine*, vol. 36, no. 4, pp. 8-17, 2021.

[17] BSI – Technical Guideline, "Cryptographic Mechanisms: Recommendations and Key Lengths," TR-02102-1, February 2, 2024.

[18] F. Lázaro, R. Raulefs, H. Bartz and T. Jerkovits, "VDES R-Mode: Vulnerability analysis and mitigation concepts," *International Journal of Satellite Communications and Networking*, vol. 41, no. 2, pp. 178-194, March/April 2023.

[19] H. Krawczyk, "Cryptographic Extraction and Key Derivation: The {HKDF} Scheme," Cryptology {ePrint} Archive, Paper 2010/264, https://eprint.iacr.org/2010/264. [Online].

[20] SkAI Data Services and Zurich University of Applied Sciences, “Live GPS Spoofing and Jamming Tracker Map,” [Online]. Available: https://spoofing.skai-data-services.com/. [Accessed 19 10 2024].