Error-Correcting Message Authentication for 5G

Elena Dubrova
School of ICT
Royal Institute of Technology
Stockholm, Sweden
dubrova@kth.se

Mats Näslund
Ericsson Research
Ericsson AB
Stockholm, Sweden
mats.naslund@ericsson.com

Göran Selander
Ericsson Research
Ericsson AB
Stockholm, Sweden
goran.selander@ericsson.com

Karl Norrman
Ericsson Research
Ericsson AB
Stockholm, Sweden
karl.norrman@ericsson.com

ABSTRACT

Message Authentication Codes (MACs) used in today's wireless communication standards are not capable of correcting errors which may occur during transmission. If MAC verification fails, the message is typically discarded and a re-transmission is requested. Re-transmissions waste energy and increase average packet latency. Excessive re-transmissions may lead to network congestion. In this paper, we introduce a new MAC, MAC-C, which efficiently combines integrity protection with single-bit error correction and provide a detailed quantitative analysis of its security.The efficiency of MAC-C makes it a good candidate for simpler 5G radio types and use cases with constrained resources such as Machine Type Communications (MTC).

CCS Concepts

•**Security and privacy → Hash functions and message authentication codes;** *Mobile and wireless security;*

Keywords

Message authentication; data integrity; hash function; error-correction; 5G

1. INTRODUCTION

Various message authentication methods are used today at control plane of 3G and 4G 3GPP mobile networks. However, these methods are not able to correct errors which may occur during transmission. Existing Message Authentication Codes (MACs), such as keyed Hash Message Authentication Code (HMAC) [2] or Cipher Block Chaining Message Authentication Code (CBC-MAC) [3], simply recompute a MAC for the received message and compare it with the received MAC. If two MACs disagree, the message is typically discarded and a re-transmission is requested. Re-transmissions waste energy and increase average packet latency. Excessive re-transmissions may lead to network congestion. It would be advantageous to have a method which efficiently combines cryptographic authentication with error-correction, reducing the number of message re-transmissions required to pass MAC verification.

In this paper, we propose a new type of MACs, MAC-C, which are capable of correcting single-bit errors in the received message. The proposed method might potentially reduce the number of message re-transmissions required to pass MAC verification since single-bit errors can be corrected by the receiving device. This translates into lower average packet latency and savings of energy.

Note that error-detection and correction always requires the addition of redundant information to the original data. i.e. increasing bandwidth, with the related consequence of spending more power for message transmission. However, if we replace the 24-bit Cyclic Redundancy Check (CRC) checksum at the physical (PHY) layer of LTE by the presented MAC-C, then we do no longer need the 32-bit message authentication code MAC-I at the Packet Data Convergence Protocol (PDCP) layer, given that MAC-C provides the same 32-bit security as MAC-I. Bandwidth can even be saved if the presented MAC can provide 32-bit security with less than the total number of bits in CRC and MAC-I. In this paper, we make a detailed quantitative analysis of the security of MAC-C and show that savings of bandwidth are possible. We also show that MAC-C guarantees the detection of the same types of errors as CRC and therefore the replacement does not impact reliability.

MAC-C seems to be particularly useful for simpler 5G radio types, e.g. the ones used for direct communication in sensor networks, and use cases such as Machine Type Communications (MTC) where storage, computing and energy resources are very constrained. It is quite likely that Forward Error Correction (FEC) codes will not be used in at least some of such simpler types of 5G radio due to the large overhead of FEC. In addition, MAC-C might potentially be useful for 3GPP mobile broadband radio access systems which are designed to adapt FEC rate to channel error conditions. For example, adaptive modulation and coding adds more error-correction bits per packet when there are higher

error rates in the channel, or takes them out when they are not needed. MAC-C may be employed in such systems in order to protect transmissions when FEC is switched off, or when the correction of single-bit errors is sufficient. In the case when FEC is switched on, the error-correcting stage of MAC-C can be switched off and MAC-C can be used for detecting errors only. However, a number of issues needs to be resolved to make MAC-C suitable for the 5G mobile broadband. We discuss these issues in Section 8.

Note that, in this paper, by "5G radio" we mean future radio types in a broad sense, not only radio specified by 3GPP standard. This is because it is expected that in the future Networked Society the wireless communication of many different types will be necessary to satisfy growing needs of society and industry.

The paper is organized as follows. Section 2 gives an introduction to linear codes and cryptographic hash functions. Section 3 introduces a new family of cryptographic hash functions. Section 4 shows how the new family of hash functions can be used for secure authentication of messages. Section 5 presents the security analysis. Section 6 analyses the class of linear codes induced by the new hash family. Section 7 describes implementation details. Section 8 discusses applicability of the presented method to 5G mobile broadband. Section 9 reviews previous work. Section 10 concludes the paper.

2. BACKGROUND

In this section, we give a brief introduction to linear codes and cryptographic hash functions based on random matrices. A reader familiar with these notions can skip this section.

2.1 Linear Codes

A (k, m) linear code over the field $GF(2)$ is a m-dimensional subspace of a vector space V_k over $GF(2)$ [19].

All codewords of a (k, m) linear code can be written as a linear combination of m basis vectors $\{v_0, v_1, \ldots, v_{m-1}\}$ as

$$c = d_0 v_0 + d_1 v_1 + \ldots + d_{m-1} v_{m-1}$$

where $d = (d_0, d_1, \ldots, d_{m-1})$ is m-bit data and "+" is the $GF(2)$ addition.

The encoding of linear codes is performed using the *generator matrix* G which is an $m \times k$ matrix whose rows are the basis vectors $\{v_0, v_1, \ldots, v_{m-1}\}$. A codeword $c = (c_0, c_1, \ldots, c_{k-1})$ is computed by multiplying the data vector d by the generator matrix G:

$$c = d \cdot G.$$

Separability is a desirable feature of codes since, after error correction, the data can be easily retrieved by truncating the last $(k-m)$ bits of a codeword. A separable linear code can be constructed by choosing basis vectors which result in a generator matrix of the form $[I_m A]$, where I_m is an identity matrix of size $m \times m$.

Errors are detected using the *parity check matrix* P which is a $(k-m) \times k$ matrix constructed as $P = [A^T I_{k-m}]$ where A^T denotes the transpose of A. The matrix P has the property that, for any codeword c,

$$P \cdot c^T = 0.$$

The encoded data can be checked for errors by multiplying it by the parity check matrix:

$$\delta = P \cdot c^T.$$

The resulting $(k-m)$-bit vector δ is called the syndrome. If the syndrome is zero, it is assumed that no error has occurred. Otherwise, the message contains an error. Depending on the code distance of the code, this error may be correctable or not.

Code distance of a code C, denoted by C_d is the minimum Hamming distance between any two distinct pairs of codewords of C. The code distance determines the error-detecting and correcting capabilities of a code. A code can correct all errors in c bits and simultaneously detect up to d additional errors if and only if its code distance is

$$C_d \geq 2c + d + 1.$$

A linear code with the code distance C_d for m-bit data can be constructed by selecting a parity check matrix P in which every subset of $C_d - 1$ columns are linearly independent.

If a code has code distance 3, then, if the syndrome δ matches one of the columns of P, it is assumed that a single-bit error has occurred. The position of the matching column in P corresponds to the bit position of the error, so the error can be located and corrected. If the syndrome is none-zero and it is not matching any of the columns of P, then it is assumed that a multiple-bit error has occurred.

2.2 Cryptographic Hash Functions based on Random Matrices

A *cryptographic hash function* is a hash function that generates a cryptographically secure hash value for an arbitrary message. Any change (accidental or intentional) to the message is expected to change the hash value, at least with a certain high probability. Thus, the hash value can be used for providing integrity assurance of the message.

Let $M = (M_0, M_1, \ldots, M_{m-1})$ be a binary message of length m. According to Carter and Wegman [26], the family of hash functions $H = \{h_A : \{0,1\}^m \to \{0,1\}^n\}$ computed as

$$h_A(M) = M \cdot A,$$

where A is an $m \times n$ random binary matrix, is a universal$_2$ family of hash functions. A family of hash functions is called *universal$_2$* (or $(1/R)$-*balanced*) if no pair of distinct inputs collide under more than $(1/R)$th fraction of the functions, where R is the size of the range of H. For $R = 2^n$, we get:

$$\forall M_1, M_2 \in \{0,1\}^m, M_1 \neq M_2 : \Pr[h_A(M_1) = h_A(M_2)] \leq \frac{1}{2^n}$$

where the probability is taken over h_A chosen uniformly at random from H.

Furthermore, the affine version of the family H, namely

$$h_{A,z}(M) = M \cdot A \oplus z,$$

where $z \in \{0,1\}^n$ is a random pad and "$\oplus$" is the bitwise XOR, is strongly universal$_2$. A family of hash functions is called *strongly universal$_2$* if the probability that any pair of distinct inputs hash to any pair of hash values is $(1/R^2)$. For the domain $\{0,1\}^m$ and range $\{0,1\}^n$, we get

$$\forall M_1, M_2 \in \{0,1\}^m, M_1 \neq M_2, \forall a_1, a_2 \in \{0,1\}^n \\ \Pr[h_A(M_1) = a_1 \wedge h_A(M_2) = a_2] \leq \frac{1}{2^{2n}}.$$

Strong universality implies *uniformity*, i.e. that all hash values are equally likely:

$$\forall M \neq 0, \forall a \in \{0,1\}^n : \Pr[h_A(M) = a] \leq \frac{1}{2^n}.$$

Krawczuk [16] has shown that if, instead of using random matrices, a pseudo-random matrix formed by the consecutive states of an n-bit LFSR with an irreducible generator polynomial is used, then the description size of Carter-Wegman family can be reduced from $n(m+1)$ to $3n$ (n bits for the generator polynomial, n bits for the initial state of the LFSR and n bits for the random pad z). The resulting hash function family is ϵ-balanced for $\epsilon = m/2^{n-1}$ instead of being perfectly balanced as Carter-Wegman family.

3. NEW HASH FUNCTION FAMILY

Our construction restricts the pseudo-random matrices even further in order to take advantage of error-correcting properties of linear codes. Similarly to Krawczuk, we use matrices formed by the consecutive states of an LFSR. In addition, we require that every row in a matrix is distinct and has the Hamming weight larger than 1. In order to assure these two properties, we encode the states of an LFSR with an even parity code and initialize the LFSR to a non-zero state. We use LFSRs in the *Fibonacci* configuration in which the the feedback is applied to the input bit only and the remaining bits shift the content of the register.

Let $(s_0, s_1, \ldots, s_{n-2})$ be an initial state of an $(n-1)$-bit LFSR. Then, in our construction, the matrix A is of type:

$$A = \begin{bmatrix} s_0 & s_1 & \ldots & s_{n-2} & c_0 \\ s_1 & s_2 & \ldots & s_{n-1} & c_1 \\ & & \ldots & & \\ s_{m-1} & s_m & \ldots & s_{m+n-3} & c_{m-1} \end{bmatrix} \quad (1)$$

where c_i is an even parity check bit for the row i.

If an LFSR is initialized to a non-zero state, then any of its states has the Hamming weight at least 1. Since an even parity check bit c_i is equal to 1 for any state $(s_i, s_{i+1}, \ldots, s_{i+n-2})$ of the Hamming weight exactly 1, any n-bit vector $(s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i)$ constructed in this way has the Hamming weight at least 2.

In order to assure that each row of an $m \times n$ matrix A is distinct, we use $(n-1)$-bit LFSRs with primitive generator polynomials. This guarantees that the period of the LFSR is maximum, i.e. $2^{n-1}-1$, implying that each non-zero state repeats not earlier than after $2^{n-1}-1$ time steps. In this way, we can generate a matrix A with up to $2^{n-1}-1$ distinct rows. Therefore, in our construction, we require the message length to be $m \leq 2^{n-1}-1$ or, equivalently, the hash length to be

$$n \geq \lceil log_2(m+1) \rceil + 1.$$

To summarize, we introduce a new hash function family $\mathcal{H} = \{h_{p,s} : \{0,1\}^m \rightarrow \{0,1\}^n\}$ where $h_{p,s}$ is computed as follows.

DEFINITION 1. *Let $p(x)$ be a primitive polynomial of degree $n-1$ over $GF(2)$. Let $s = (s_0, s_1, \ldots, s_{n-2})$ be an initial state of an $n-1$-bit LFSR in the Fibonacci configuration with the generator polynomial $p(x)$. For each $p(x)$ and each $s \neq 0$, we associate a hash function $h_{p,s}$ such that, for any binary message M of length $m \leq 2^{n-1}-1$, $h_{p,s}(M)$ is defined as the linear combination*

$$h_{p,s}(M) = \sum_{i=0}^{m-1} M_i \cdot (s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i), \quad (2)$$

where $c_i = \sum_{j=i}^{i+n-2} s_j$ is an even parity check bit and the addition and the multiplication are carried out in $GF(2)$.

The affine version of the family $\mathcal{H}$ is defined as $\mathcal{H}_{\mathcal{A}} = \{h_{p,s,z} : \{0,1\}^m \rightarrow \{0,1\}^n\}$ where $h_{p,s,z}(M)$ is given by

$$h_{p,s,z}(M) = h_{p,s}(M) \oplus z, \quad (3)$$

where $z \in \{0,1\}^n$ is a random pad and "$\oplus$" is the bitwise XOR.

The difference between Krawczuk's hash family [16] and the hash family $\mathcal{H}$ is that we require the polynomial $p(x)$ to be primitive rather than irreducible and that we use the nth bit of the hash function for the parity check. As a result, for $m \leq 2^{n-1}-1$, all m vectors $(s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i)$ in the linear combination (2) are distinct and have the Hamming weight larger than 1. As we show in Section 6, this implies that the class of linear codes induced by $\mathcal{H}$ has the code distance 3 and hence it can correct single-bit errors.

The description size of the hash family $\mathcal{H}_{\mathcal{A}}$ is $3n-2$ ($n-1$ bits for the generator polynomial, $n-1$ bits for the initial state, 1 bit for the parity bit and n bits for the random pad).

Since we use only a subset of random matrices for constructing the hash functions, the resulting family is only ϵ-balanced for a small ϵ (evaluated in Section 5) rather than perfectly balanced as Carter-Wegman family. For the purpose of authentication, this small ϵ represents no substantial loss, while the guaranteed error-correction properties make the resulting hash function family significantly more interesting.

4. MESSAGE AUTHENTICATION

In this section, we show how the presented family of hash functions $\mathcal{H}_{\mathcal{A}}$ can be used for secure authentication of messages.

We assume a typical setting [21] in which the sender and the receiver transmit messages over an insecure channel where messages can be maliciously modified, e.g. a public 5G radio network. The sender and the receiver share a secret key, i.e. a shared secret which is unknown to the adversary. The shared secret can be established, for example, by public key techniques or symmetric techniques supported by SIM or USIM cards, or similar, using the traditional methods [21].

A *message authentication* algorithm accepts as input a secret key and a message to be authenticated and outputs an *authentication tag*. The tag allows legitimate users, who possess the secret key, to detect any changes in the message content.

A sender authenticates a message M by computing the authentication tag MAC-C as $t = h_{p,s,z}(M)$, where $h_{p,s,z}(M)$ is defined by (3) and the primitive polynomial $p(x)$, the initial state s and the pad z are selected pseudo-randomly based on a secret key. The computed tag t is appended to M and the result is transmitted.

The modification of the linear hash function (2) to the affine one (3) is necessary to prevent an attacker from injecting all-0 messages. Without such a modification, the hash value of an all-0 message would always be 0, independently of the polynomial $g(x)$ and the initial state s. The reader familiar with e.g. the UIA2 MAC of the 3G standard will recognize this type of construction. In that case, the pad z is generated by the SNOW3G stream cipher [1].

A receiver authenticates a received message M' (which may potentially differ from the submitted message M) by

re-computing the tag for M', $t'' = h_{p,s,z}(M')$, and verifying if the received tag t' and the re-computed one t'' are the same:

$$\delta = t'_{p,s,z} \oplus t''_{p,s,z}.$$

A non-zero syndrome δ indicates an error. Single-bit errors can be corrected by checking if the δ matches one of the columns of the parity check matrix P of the linear code induced by the presented construction (explained in Section 7). If the δ matches one of the columns of P, the corresponding bit in the received message is complemented.

5. SECURITY ANALYSIS

In this section, we analyze the security of MAC-C.

5.1 Detection of Malicious Errors

First, we derive a bound on the probability of not detecting an error crafted by an adversary, i.e. the probability that, after observing a message M and its tag t, the adversary can find M' and t' such that $M' \neq M$ and t' is a valid tag for M'. In this case, the error crafted by the adversary will not be detected.

The following Theorem was proven in [16].

THEOREM 1. *[16] Let $p(x)$ be an irreducible polynomial of degree n over $GF(2)$ and let $s = (s_0, s_1, \ldots, s_{n-1})$ be an initial state of the LFSR with the generator polynomial $p(x)$. Let M be an m-bit message. Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the n different roots of $p(x)$ over $GF(2^n)$. Then*

$$h_{p,s}(M) = BD_{M,p}B^{-1}s$$

where B is a non-singular $n \times n$ matrix which depends on $p(x)$ only and $D_{M,p}$ is an $n \times n$ diagonal matrix with $M(\lambda_i)$, $1 \leq i \leq n$, as its diagonal entry.

Any primitive polynomial is irreducible. Furthermore, any primitive polynomial of degree n has n different roots over $GF(2^n)$ which all have order $2^n - 1$ [18]. Therefore, Theorem 1 holds for the case when $p(x)$ is a primitive polynomial of degree n over $GF(2)$ as well.

From this we can derive the following property of the hash family $\mathcal{H}$.

THEOREM 2. *For any hash function $h_{p,s}$ chosen uniformly at random from the family $\mathcal{H}$,*

$$\forall M \neq 0, \forall a \in \{0,1\}^n : \ Pr[h_{p,s}(M) = a] \leq \frac{m}{\phi(2^{n-1} - 1)}, \tag{4}$$

where ϕ is the totient function.

Proof: First, we observe that the truncation of the parity check bit does not change the probability $Pr[h_{p,s}(M) = a]$. Let $h'_{p,s}$ be an $n-1$ bit hash value obtained by truncating the parity check bit of $h_{p,s}$. In other words, $h'_{p,s}$ is computed as

$$h'_{p,s}(M) = \sum_{i=0}^{m-1} M_i \cdot (s_i \ s_{i+1} \ldots s_{i+n-2}).$$

for each primitive polynomial $p(x)$ of degree $n-1$ over $GF(2)$ and each initial state $s \neq 0$. Then $Pr[h_{p,s}(M) = a] = Pr[h'_{p,s}(M) = a]$.

Next we estimate the probability that $h'_{p,s}(M) = a'$ for any $a' \in \{0,1\}^{n-1}$, for a randomly chosen primitive polynomial $p(x)$ and an initial state $s \neq 0$.

Fix a message $M \neq 0$ and a'. We use the fact that the polynomial $M(x)$ corresponding to the message M has a common root with $p(x)$ if and only if $p(x)$ divides $M(x)$. Consider two cases according to the value a'.

Case 1: Let $a' = 0$. Since $s \neq 0$, then $h'_{p,s}(M) = 0$ only if $D_{M,p}$ is singular, since the matrices B and B^{-1} are not singular. This can happen only if for some i, $M(\lambda_i) = 0$ or, equivalently, only if $p(x)$ divides $M(x)$. The probability for this to happen is at most the number of possible primitive factors of $M(x)$ divided by the total number of primitive polynomials of degree $n-1$.

Because of the unique factorization property, there are at most $m/(n-1)$ primitive factors of $M(x)$ each of degree $n-1$. On the other hand, the total number of primitive polynomials of degree $n-1$ over $GF(2)$ is $\phi(2^{n-1}-1)/n-1$. Therefore

$$Pr[h'_p(M) = 0] \leq \frac{m}{\phi(2^{n-1} - 1)}.$$

Case 2: Let $a' \neq 0$. Then $h'_{p,s}(M) = a'$ only if $D_{M,p}$ is non-singular and s is the unique vector which is mapped by $BD_{M,p}B^{-1}$ into a'. The vector s assumes this value with probability of $1/(2^{n-1} - 1)$, and therefore $Pr[h'_{p,s}(M) = a'] \leq 1/(2^{n-1} - 1)$ for $a' \neq 0$.

In both cases, $Pr[h'_{p,s}(M) = a'] \leq \frac{m}{\phi(2^{n-1}-1)}$.

□

The totient function $\phi(k)$, also called Euler's totient function, is defined as the number of positive integers less than or equal to k that are relatively prime to k (i.e. not containing any factor in common with k):

$$\phi(k) = \{\#n \mid n < k \wedge gcd(k, n) = 1\},$$

where *gcd* stands for the greatest common divisor.

It is known that, if k is prime, then $\phi(k) = k-1$. It is also known that, if k is of type k^a where k is prime and $a > 0$, then $\phi(k^a) = k^a - k^{a-1}$. Another property which we use is $\phi(2^a - 1) \geq \phi(2^a)$.

If $2^{n-1} - 1$ is prime, then (4) reduces to

$$Pr[h_{p,s}(M) = a] \leq \frac{m}{2^{n-1} - 2}.$$

Since $\phi(2^{n-1} - 1) \geq \phi(2^{n-1}) = 2^{n-2}$, for a general case, we can approximate (4) as:

$$Pr[h_{p,s}(M) = a] \leq \frac{m}{2^{n-2}}. \tag{5}$$

5.2 Correction of Malicious Errors

An adversary may attempt to craft a multiple-bit error which appears as a single-bit error to our message authentication algorithm. In this case, the algorithm will be tricked to correct this error and accept the message after correction as legitimate.

Such a situation may happen only if, after observing a message M and its tag t, the adversary can find M' and $t'' = t' + v$ where t' is a valid tag for M' and v is one of the m vectors $(s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i)$ in the linear combination (2) used to compute t'.

From the proof of Theorem 2, we know that $Pr[h_{p,s}(M) = 0] \leq \frac{m}{\phi(2^{n-1}-1)}$ and $Pr[h_{p,s}(M) = a] \leq 1/(2^{n-1} - 1)$ for all $a \neq 0$. Since none of m vectors $(s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i)$ in the linear combination (2) repeats more than once, we can conclude that the probability that an adversary succeeds to

craft a multiple-bit error which appears as a single-bit error to our message authentication algorithm is at most ϵ_c, where ϵ_c is given by

$$\epsilon_c \leq \frac{m}{\phi(2^{n-1}-1)} + \frac{m-1}{2^{n-1}-1}. \quad (6)$$

If $2^{n-1}-1$ is prime, then (6) reduces to

$$\epsilon_c \leq \frac{m}{2^{n-1}-2} + \frac{m-1}{2^{n-1}-1} \leq \frac{2m-1}{2^{n-1}-2}.$$

For a general case, we can approximate (6) as:

$$\epsilon_c \leq \frac{m}{2^{n-2}} + \frac{m-1}{2^{n-1}-1} \leq \frac{3m-1}{2^{n-1}-1}. \quad (7)$$

6. ANALYSIS OF RANDOM ERRORS

In this section, we analyze properties of linear codes induced by the presented construction.

It is easy to show that an $(m+n, m)$ linear code with the generator matrix $G = [I_m A]$ where A is of type (1) has the code distance of at least 3. If A is of type (1), then the parity check matrix $P = [A^T I_n]$ of the code does not contain any 0 columns and every column of P is distinct. This implies that every pair of columns of P is linearly independent and thus the code distance is at least 3.

A code with the code distance 3 can correct single-bit errors. Furthermore, since the matrix A of type (1) contains a parity check bit, such a code is capable of detecting all errors affecting an odd number of bits. Next, we show that it can also detect all burst errors of length $n-1$ or smaller. A *burst* error is an error affecting adjacent bits. Burst errors are a dominant type of errors in data communication and storage.

THEOREM 3. *An $(m+n, m)$ linear code with the generator matrix $G = [I_m A]$, where A is of type (1) detects all burst errors of length $n-1$ or smaller.*

Proof: Suppose a burst error e of length $n-1$ or smaller occurred in a message M, i.e. the received message is of type $M'(x) = M(x) + e(x)$, where $e(x)$ is the polynomial corresponding to the bit string e. Then the re-computed check bits are given by the hash value

$$\begin{aligned} h_{p,s}(M') &= \sum_{i=0}^{m-1} (M_i + e_i) \cdot (s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i) \\ &= h_{p,s}(M) + \sum_{i=0}^{m-1} e_i \cdot (s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i). \end{aligned}$$

The error e will not be detected only if

$$\sum_{i=0}^{m-1} e_i \cdot (s_i, s_{i+1}, \ldots, s_{i+n-2}, c_i) = 0. \quad (8)$$

Following the same reasoning as in the proof of Theorem 2, we can conclude that the equation (8) holds only if $D_{e,p}$ is singular. This can happen only if for some i, $e(\lambda_i) = 0$. The error polynomial $e(x)$ has a common root with $p(x)$ if and only if $p(x)$ divides $e(x)$. This is not possible if the degree of $e(x)$ is smaller than the degree on $p(x)$.

Since the error e is of length $n-1$ or smaller, the degree of $e(x)$ is $n-2$ or smaller. The degree of $p(x)$ is $n-1$. Therefore, the error e will be detected.

□

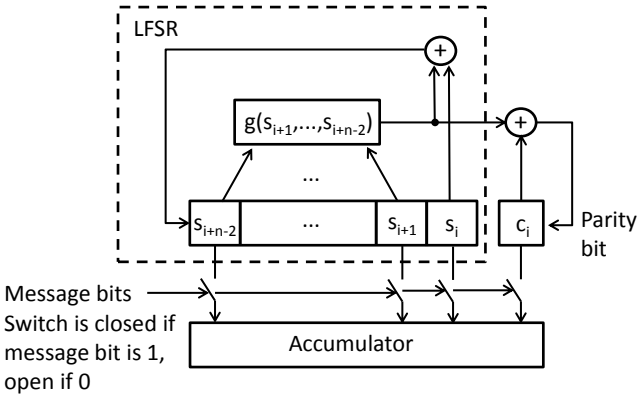

Figure 1: The computation of the hash function $h_{p,s}(M)$ for a message M.

7. IMPLEMENTATION DETAILS

In this section, we show how MAC-C can be computed efficiently.

7.1 Encoding/Decoding

The computation of the hash function $h_{p,s}(M)$ for a message M can be efficiently implemented as shown in Figure 1. The LFSR is initialized to the state s and the parity bit computed for s. The LFSR advances its state which each message bit. If the message bit is 1, the corresponding LFSR state with its parity bit is accumulated into the accumulator register. Otherwise, the state is not accumulated. The accumulator adds its input vector to the vector of its current state, where the addition is the bitwise XOR. After m time steps, the accumulator register contains $h_{p,s}(M)$.

Usually the generator polynomial of an LFSR is fixed and therefore the circuit implementing it has hardwired feedback connections. Implementation of hash functions in the presented method requires an LFSR in which the feedback connections are programmable (and re-programmable), since we should be able to change generator polynomials. Note that some LFSRs used in non-cryptographic applications also may use programmable connections, e.g. LFSRs implementing CRC encoding which need to support different CRC standards based on different generator polynomials [5].

The generator polynomial $p(x)$ and the initial state s should be changed periodically in such a way that the matrix M which they induce appears randomly selected for an adversary. In general, it is sufficient to compute a new generator polynomial and/or a new initial state at the beginning of each session, and keep them fixed for all messages. The pad z, however, has to be changed for every message.

Note that if encryption using a stream cipher is applied at sender, pad z may be provided by the encryption function, thus "interleaving" encryption and authentication processing. In this case, receiver may either (i) first remove pad z by decrypting and then treat only $h_{g,s}(M)$ as tag, or (ii) not remove pad z and treat $h_{g,s}(M) \oplus z$ as tag.

7.2 Error-Correction

Single-bit errors can be corrected by checking if the syndrome δ matches one of the columns of the $n \times (m+n)$ parity check matrix $P = [A^T I_n]$ of the linear code induced by the

presented family of hash functions. If A is of type (1) then A^T is of type:

$$A = \begin{bmatrix} s_0 & s_1 & \dots & s_{m-1} \\ s_1 & s_2 & \dots & s_m \\ & & \dots & \\ s_{n-2} & s_{n-1} & \dots & s_{m+n-3} \\ c_0 & c_1 & \dots & c_{m-1} \end{bmatrix} \quad (9)$$

Thus, m columns of parity check matrix P correspond to the m consecutive states of the LFSR starting from the initial state s with the parity check bit appended:

$$(s_i, s_{i+1}, \dots, s_{i+n-2}, c_i)$$

for $i = \{0, 1, \dots, m-1\}$.

Therefore, we can check if the syndrome δ matches one of these states by initializing the LFSR to the state s and the 1-bit register to the parity bit of s, then clocking the LFSR at most m times and, at each step i, comparing the n-bit syndrome δ to the n-bit vector $(s_i, s_{i+1}, \dots, s_{i+n-2}, c_i)$:

$$r_i = \delta \oplus (s_i, s_{i+1}, \dots, s_{i+n-2}, c_i)$$

for $i = \{0, 1, \dots, m-1\}$, where "$\oplus$" is the bitwise XOR. As soon $r_i = 0$ for some i, the process stops and the ith bit of the received message M' is complemented. A counter can be used to keep track of the position of the erroneous bit i.

7.3 Computation of parity bit

Encoding of LFSR states in an even parity code can be implemented using a single 2-input XOR gate and an additional 1-bit register as follows.

Let c_i be the variable representing the value of 1-bit register which stores the parity check bit of a current state $(s_i, s_{i+1}, \dots, s_{i+n-2})$ of an $(n-1)$-bit LFSR. At each clock cycle, the value of the input bit of the LFSR is updated using the feedback function

$$f(s_i, s_{i+1}, \dots, s_{i+n-2}) = s_i + g(s_{i+1}, s_{i+2}, \dots, s_{i+n-2}) \quad (10)$$

where $g(s_{i+1}, s_{i+2}, \dots, s_{i+n-2})$ is determined by the generator polynomial of the LFSR and the rest LFSR's bits update their values as a shift from the previous bit. So, the parity of the next state, c_{i+1}, can be computed as

$$c_{i+1} = c_i \oplus s_i \oplus f(s_i, s_{i+1}, \dots, s_{i+n-2}). \quad (11)$$

From (10) we can conclude that $s_i \oplus f(s_i, s_{i+1}, \dots, s_{i+n-2}) = g(s_{i+1}, s_{i+2}, \dots, s_{i+n-2})$ and therefore (11) an be re-written as

$$c_{i+1} = c_i \oplus g(s_{i+1}, s_{i+2}, \dots, s_{i+n-2}). \quad (12)$$

So, to implement an even parity code, the 1-bit register which stores the parity check bit of current state should be updated at each clock cycle according to (12).

Note that the above construction assumes that, during the initialization of the LFSR, the 1-bit register which stores the parity check bit of the initial state is also initialized to a correct pre-computed value. In other words, we require that binary n-tuples which are used for the initialization of the LFSR and the 1-bit register which stores the parity check bit are drawn pseudo-randomly from the set of all possible binary vectors of length n which have even Hamming weight.

8. SUITABILITY FOR 5G

In this section, we analyze pros and cons of two cases of implementing integrity protection in 5G. We mainly focus on LTE standard because, due to its relative complexity compared to simpler standards like ZigBee, it brings more issues. The two cases we consider are:

A) Using a 32-bit message authentication code MAC-I at the PDCP layer, and

B) Using a MAC-C at the PHY layer.

Note that there are two corresponding cases for ZigBee [14], where either Application Support Sublayer (APS), or Network Layer (NWK), performs integrity protection using a 32-bit *Message Integrity Code* (MIC) similarly to the PDCP layer of LTE. In ZigBee, the CRC is called *Frame Check Sequence* (FCS) and it is 16-bit.

We first discuss implications of applying integrity protection at the PHY layer. Then, we estimate the size of MAC-C required to get 32-bit security. Finally, we evaluate impact on bandwidth.

Note that, in LTE, MAC-I is used for the control plane signalling only. Data integrity at the user plane is not provided. This may change during the development of the 5G. With radio access as a building block in, for example, industrial automation, traffic control, smart grid, and e-health, adding integrity protection to the user plane might become a necessity.

8.1 Integrity Protection at the PHY Layer

We now analyze the architectural implications of applying integrity protection at the PHY layer. In particular, starting from the fact that, in LTE, PDCP provides integrity protection, we investigate implications for security caused by instead providing it at the PHY layer.

The protocol architecture in LTE places the security policy control in the Radio Resource Control (RRC) layer. RRC starts security, changes keys, selects encryption algorithms etc. The security policy enforcement is provided by PDCP in the form of integrity protection and encryption. In the original LTE architecture, RRC and PDCP is located in the same node. More specifically, they are located in the same secure environment.

With the introduction of the Dual Connectivity feature, the protocol stack is allowed to be split in two parts. For the control plane, all layers of the stack remain in the controlling part of the base station (eNB). However, for the user plane, it is possible to locate the PDCP, Radio Link Control (RLC), Medium Access Control (MAC) and PHY layers in a separate part of the eNB, here called the *controlled* part. The controlled part of the eNB may be physically separated from the controlling part. As a result, RRC needs to provide key material to the PDCP entities in the controlled part; this must be done over a confidentiality and integrity protected link. Further, since the PDCP entities are located in the controlled part of the eNB, they must also be enclosed in a secure environment.

We assume a similar split in the 5G protocol architecture. Applying integrity protection on the PHY layer then have similar consequences as for the Dual Connectivity feature. In fact, only considering the user plane, exactly the same security architecture can be used. Considering the control plane, the trust model must be altered under some circumstances though.

Should 5G require that RRC is integrity protected between the User Equipment (UE) and the controlling part

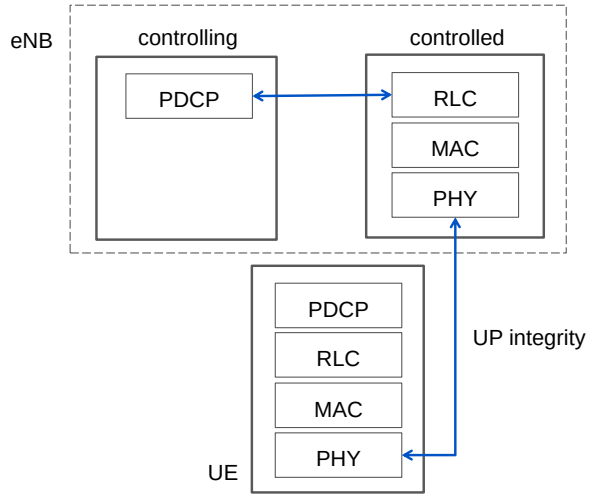

Figure 2: Assumed protocol and security architecture for 5G user plane (UP).

MAC-C length n, bits	Message length m, bits	Failure probability	
		Error Detection	Error Correction
40	43	$2^{-32.6}$	2^{-32}
41	85	$2^{-32.6}$	2^{-32}
42	171	$2^{-32.6}$	2^{-32}
43	341	$2^{-32.6}$	2^{-32}
44	683	$2^{-32.6}$	2^{-32}
45	1365	$2^{-32.6}$	2^{-32}
46	2731	$2^{-32.6}$	2^{-32}
47	5461	$2^{-32.6}$	2^{-32}
48	10923	$2^{-32.6}$	2^{-32}
49	21864	$2^{-32.6}$	2^{-32}
50	43692	$2^{-32.6}$	2^{-32}
51	87384	$2^{-32.6}$	2^{-32}
52	174768	$2^{-32.6}$	2^{-32}
53	349536	$2^{-32.6}$	2^{-32}
54	699072	$2^{-32.6}$	2^{-32}

Table 1: Values of n and m required to provide 32-bit security by MAC-C.

via the controlled part, applying integrity on the physical layer implies that the protection can only be provided hop-by-hop. That is, the RRC protocol is no longer protected end-to-end between the UE and the controlling part. This is the case even if the PDCP entity were located in the controlling part (see Fig. 2). In fact, applying the integrity protection on the PHY layer makes end-to-end protection impossible in this setting. Moreover, this is a trust model in which an operator needs to trust more entities compared to LTE; the processing environment of the controlled part and the protection of the link between it and the controlling part must be trusted. Whether this trust model is applicable or not, is heavily dependent on the trust one can put on actual implementations of the secure environments in controlled parts. It should be noted that the controlled parts may for 5G not be full base stations, but could be cheaper radio-heads. Such devices may become too expensive to be commercially viable if there are too strict requirements on their implementation. Bearing in mind that a breach of the control plane security can be much more severe than a breach of the user plane security, integrity protection at the PHY layer might be not suitable for the control plan.

To summarize, the implications for security caused by providing integrity protection at the PHY layer seem to indicate that, while integrity protection on the PHY layer is appropriate for the user plane of LTE, it may not fit the control plane.

ZigBee standard supports multi-hop, i.e. packets can be delivered from one node to another passing through multiple radio hops. In this case, implementing integrity protection on the PHY layer of ZigBee raises problems similar to LTE. However, ZigBee also supports direct communication. In the direct communication mode, the FCS and the MIC are checked by the same receiving node, enabling end-to-end security also for the case when integrity protection is performed on the PHY layer.

8.2 Soft Combining

In LTE, the actions taken to detect malicious and non-malicious faults differ. Malicious fault detection is performed at the PDCP layer. If MAC-I verification fails, the PDCP packet is not further processed.

Non-malicious fault detection is performed at the PHY layer. If CRC check fails, the transport block may be kept together with other re-transmitted versions for later processing; this is called *soft combining*. Soft combining may successfully restore a transport block from two or more transport blocks with failed CRCs.

There are two ways to manage this situation when using a MAC-C at the PHY layer that detects both malicious and non-malicious faults. First, taking a pure security stance, one may propose that any transport block with a failed MAC-C shall be discarded. This implies that soft combining cannot be used and it may not acceptable from a performance perspective. Second, one may propose that the MAC-C of the reconstructed transport block shall be verified again. It is important that the chosen way of reconstructing transport blocks does not open up possibilities for an attack. Consequences of reconstruction on security have to be further explored.

ZigBee standard does not support soft combining, so this problem does not arise. As described in the paragraph 5.2.1.9 of [14], MAC layer discards all the received frames for which the re-computed and the received CRC values disagree.

8.3 Replay Protection

Moving the integrity protection to the PHY layer also has implications for replay protection. In LTE, replay protection is performed by a receiver at the PDCP layer. The receiver of the PDCP packet essentially determines whether the sequence number in the PDCP header has been received previously. If that is the case, the packet is considered a replay and is not further processed.

If we do integrity checking to the PHY layer, we no longer have access to the sequence number in the PDCP packet header. Even though the sequence number may be visible at the PHY layer it would be a gross layer violation to access it. Moreover, due to the fact that a PDCP packet can get

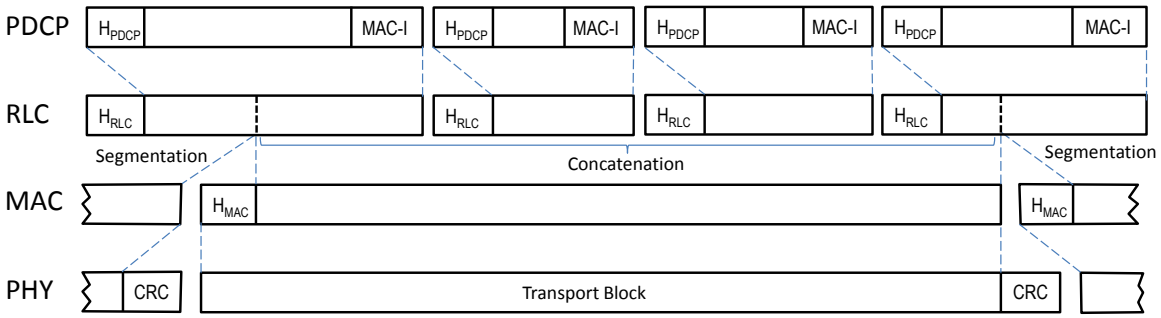

Figure 3: Data flow through protocol layers of LTE; $\mathbf{H}_X$ stands for the header of layer X.

segmented into multiple transport blocks (segmentation is explained in more details in Section 8.5), each carrying its own MAC-C, it is not guaranteed that each transport block carries a PDCP sequence number that can be used for replay protection.

To resolve this issue, one could instead base the replay protection on the RLC sequence number which is included in each transport block. The RLC sequence number should then be extended with an overflow counter similarly to the PDCP sequence number.

Similarly, in ZigBee, replay protection can be implemented based on sequence numbers of NWK layer included in each transport block.

8.4 Required Size of MAC-C

We have computed values of MAC-C length, n, and message length, m, required to obtain 32-bit security provided by the MAC-I or MIC. Table 1 shows the results. The 3rd column shows the probability of not detecting an error computed according to (5). The 4th column shows the probability of correcting wrongly an error computed according to (7). For a given n in column 1, for all messages of length smaller than m in column 2, the security higher than 32 bits is guaranteed in both, error-detection and error-correction cases.

In LTE, the maximum transport block size is 75k bits in 3GPP Release 8 and 391k bits in 3GPP Release 13 [11]. Assuming that the size of the maximum transport blocks used in 5G mobile broadband will not exceed 699k bits, a 54-bit MAC-C is sufficient to provide 32-bit security of MAC-I.

In ZigBee, the maximum PHY Service Data Unit (PSDU) size is 2014 bytes (16112 bits) for Smart Utility Networks (SUN) PHY layer and 127 bytes (1016 bits) of other PHY layers [14]. So, MAC-C of size 49 bits and 45 bits, respectively, is sufficient to provide 32-bit security of MIC.

8.5 Impact on Bandwidth

To estimate impact on bandwidth for both cases, we need to analyze the flow of data through protocol layers. Figure 3 shows the diagram of LTE protocol stack for the control plane. In the case A when a 32-bit MAC-I at the PDCP layer is used for integrity protection at the user plane, the data flow will look the same.

If the data integrity is protected at the PDCP layer, then a 32-bit MAC-I is appended to the packet coming from the layer above, as shown in Figure 3. In addition, a PDCP header is added and the result is submitted to the RLC layer.

RLC layer concatenates or segments the packets coming from PDCP layer into a size appropriate for transport blocks and forwards it to the MAC layer with its own header. Concatenation is performed when the PDCP packet is small compared to the available radio data rate (resulting in large transport blocks). Segmentation is done when the PDCP packet is large compared to the available radio data rate (resulting in small transport blocks). Both concatenation and segmentation may be performed to form a transport block, as shown in Figure 3.

MAC layer adds its own header, and performs padding to fit a packet in a transport block. Finally, MAC layer submits the result to the PHY layer.

At PHY layer, a 24-bit CRC is appended to the transport block and the block is transmitted into physical channels.

We can conclude that, in LTE, if we replace a 24-bit CRC and a 32-bit MAC-I by a 54-bit MAC-C, then the difference in bits per transport block is given by $N \cdot 32 - 30$, where N is the number of occurrences of MAC-I in the transport block. So, for all $N > 0$, we save $2 + 32 \cdot (N - 1)$ bits. The case of $N = 0$ occurs when a PDCP packet has to be splitted into two or more segments to fit the transport block and at least one of these segments occupy the complete transport block and does not contain MAC-I.

Similarly to LTE, ZigBee supports fragmentation an reassembly at the ASP layer [29], so a PSDU may potentially contain from zero to several MICs. In the case of SUNs, if we replace a 16-bit FCS and a 32-bit MIC by a 49-bit MAC-C, the difference in bits per PSDU is $N \cdot 32 - 33$, where N is the number of occurrences of MIC in PSDU. For a 45-bit MAC-C, we get $N \cdot 32 - 29$.

In conclusion, although we have given a concrete formula to calculate the bandwidth gain of the presented method, it is difficult to determine whether this gain will be substantial in practice. The main reason for this is that there are too many parameters affecting N to be able to describe a typical situation. For example, in LTE, the transport block size depends on the modulation and coding scheme and the number of resource blocks assigned to the UE. These in turn depend on distance from the eNB, quality of coverage, size of the data being transmitted, etc. Parameters vary widely over time, from user to user and depending on geographical location.

9. RELATED WORK

Message authentication codes have also been thoroughly

investigated in the past, see Simmons for an excellent survey [20].

Carter and Wegman [26] were first to show that hash functions can be combined with one-time pads to construct strong authentication algorithms. Their approach was further developed by Brassared [7], Desmedt [8] and Krawczyk [16].

Stinson [22] introduced the notion of almost strongly universal hash families which helped reduce the key size of unconditionally secure MACs proposed earlier by Gilbert et al. [13].

To our best knowledge, the use of error-correcting codes as MACs was first proposed by T. Johansson et al. in [15] and the first construction of universal families of hash functions via codes was presented by Bierbrauer et al. in [4].

The relation between error-correcting codes and hash functions has been further investigated by Stinson in [23]. Black et al. have shown that universal hash families can be applied to construct efficient computationally secure MACs, e.g. UMAC [6]. Computationally secure MACs are used in 3G wireless communication.

Building up on the work of Krawczuk [16], MACs based on BCH and Reed-Solomon error-correcting codes capable of correcting two bit errors have been presented in [17]. Although more powerful than our construction in terms of the number of corrected bits, these MACs require a heavier error-correction scheme.

Fuzzy or approximate MACs with the property that a message will pass the authenticity check if it has less than a certain small number of errors have been presented in [28, 12, 27, 24, 25]. Approximate MACs are an interesting concept and they are very valuable for applications where some noise is acceptable, e.g. multimedia (image, video, etc.) or biometrics authentication. However, they simply ignore errors rather than correct them, as the MAC presented in this paper.

A number proposals to replace the traditional CRC checksum with a cryptographically secure CRC have been made, including [16, 10, 9]. These MACs can detect random and malicious errors, but they cannot correct them.

10. CONCLUSION

In this paper, we introduced a new type of MACs, MAC-C, based on linear codes with code distance 3 and performed a quantitative analysis of the its security.

We conclude that MAC-C seems a good candidate for simpler 5G radio types, such as the ones used for direct communication in sensor networks, and use cases with constrained resources such as MTC. In these simpler types of 5G radios, FEC and soft-combining are typically not used and therefore the issues we discussed in Section 8 do not arise.

MAC-C appears less suitable for the 5G mobile broadband, unless the issues we discussed in Section 8 are resolved. We are currently working on these problems.

11. ACKNOWLEDGEMENTS

The authors would like to warmly thank Vlasios Tsiatsis from Ericsson Research for sharing his expertise on ZigBee standard.

The first author was supported by the research grant No SM14-0016 from the Swedish Foundation for Strategic Research.

12. REFERENCES

[1] 3GPP. ETSI/SAGE specification: Specification of the 3GPP confidentiality and integrity algorithms UEA2 & UIA2, document 2: SNOW 3G specification, 2006.

[2] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In N. Koblitz, editor, *Advances in Cryptology - CRYPTO'96*, volume 1109 of *LNCS*, pages 1–15. Springer Berlin Heidelberg, 1996.

[3] M. Bellare, J. Kilian, and P. Rogaway. The security of cipher block chaining. In Y. Desmedt, editor, *Advances in Cryptology - CRYPTO'94*, volume 839 of *LNCS*, pages 341–358. Springer Berlin Heidelberg, 1994.

[4] J. Bierbrauer, T. Johansson, G. Kabatianskii, and B. Smeets. *Advances in Cryptology - CRYPTO'93*, chapter On Families of Hash Functions via Geometric Codes and Concatenation, pages 331–342. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[5] J. Birch, L. G. Christensen, and M. Skov. A programmable 800 Mbit/s CRC check/generator unit for LANs and MANs. *Comput. Netw. ISDN Syst.*, 24(2):109–118, Apr. 1992.

[6] J. Black, S. Halevi, H. Krawczyk, T. Krovetz, and P. Rogaway. UMAC: Fast and secure message authentication. In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology - CRYPTO'99*, pages 216–233, London, UK, UK, 1999. Springer-Verlag.

[7] G. Brassard. On computationally secure authentication tags requiring short secret shared keys. In D. Chaum, R. Rivest, and A. Sherman, editors, *Advances in Cryptology*, pages 79–86. Springer US, 1983.

[8] Y. Desmedt. Unconditionally secure authentication schemes and practical and theoretical consequences. In H. C. Williams, editor, *Advances in Cryptology - CRYPTO'85 Proceedings*, volume 218 of *Lecture Notes in Computer Science*, pages 42–55. Springer Berlin Heidelberg, 1986.

[9] E. Dubrova, M. Naslund, and G. Selander. CRC-based message authentication for 5G mobile technology. In *Proceedings of 1st IEEE International Workshop on 5G Security*, August 2015.

[10] E. Dubrova, M. Naslund, G. Selander, and F. Lindqvist. Cryptographically secure CRC for lightweight message authentication. Technical Report 2015/035, January 2015. Cryptology ePrint Archive.

[11] ETSI. LTE evolved universal terrestrial radio access (E-UTRA); user equipment (ue) radio access capabilities, 2011. 3GPP TS 36.306, V10.2.0.

[12] R. Ge, G. Arce, and G. Di Crescenzo. Approximate message authentication codes for n-ary alphabets. *IEEE Transactions on Information Forensics and Security*, 1(1):56–67, March 2006.

[13] E. N. Gilbert, F. J. MacWilliams, and N. J. A. Sloane. Codes which detect deception. *Bell System Technical Journal*, 53(3):405–424, 1974.

[14] IEEE. Local and metropolitan area network standards: PHY/MAC layer for ZigBee, 2011. https://standards.ieee.org/getieee802/download/802.15.4-2011.pdf.

[15] T. Johansson, G. Kabatianskii, and B. Smeets.

Advances in Cryptology - EUROCRYPT'93, chapter On the Relation Between A-Codes and Codes Correcting Independent Errors, pages 1–11. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[16] H. Krawczyk. LFSR-based hashing and authentication. In *Proc. of the 14th Annual Int. Cryptology Conf. on Advances in Cryptology - CRYPTO'94*, pages 129–139, London, UK, UK, 1994. Springer-Verlag.

[17] C. C. Y. Lam, G. Gong, and S. A. Vanstone. Message authentication codes with error correcting capabilities. In *Proc. of 4th Int. Conf. on Information and Communication Security*, pages 354–366, London, UK, 2002. Springer-Verlag.

[18] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications*. Cambridge Univ. Press, 1994.

[19] W. W. Peterson and E. J. Weldon. *Error-Correcting Codes*. The MIT Press, 1st edition, 1961.

[20] G. Simmons. A survey of information authentication. *Proceedings of the IEEE*, 76(5):603–620, May 1988.

[21] D. Stinson. *Cryptography Theory and Practice*. Chapman & Hall/CRC, 3rd edition, 2006.

[22] D. R. Stinson. Universal hashing and authentication codes. *Des. Codes Cryptography*, 4(4):369–380, Oct. 1994.

[23] D. R. Stinson. On the connections between universal hashing, combinatorial designs and error-correcting codes. In *In Proc. Congressus Numerantium 114*, pages 7–27, 1996.

[24] D. Tonien, R. Safavi-Naini, P. Nickolas, and Y. Desmedt. *Coding and Cryptology: Second International Workshop, IWCC 2009, Zhangjiajie, China, June 1-5, 2009. Proceedings*, chapter Unconditionally Secure Approximate Message Authentication, pages 233–247. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[25] O. Ur-Rehman, N. Zivic, S. Tabatabaei, and C. Ruland. Error correcting and weighted noise tolerant message authentication codes. In *5th Int. Conf. on Signal Processing and Communication Systems (ICSPCS)*, pages 1–8, Dec 2011.

[26] M. N. Wegman and J. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3):265 – 279, 1981.

[27] S. Xiao and C. G. Boncelet. Efficient noise-tolerant message authentication codes using direct sequence spread spectrum technique. In *Information Sciences and Systems, 2006 40th Annual Conference on*, pages 1640–1644, March 2006.

[28] L. Xie, G. R. Arce, and R. F. Graveman. Approximate image message authentication codes. *IEEE Transactions on Multimedia*, 3(2):242–252, Jun 2001.

[29] ZigBee Aliance. Zigbee specification, 2012. Document 053474r20.