Annales UMCS Informatica AI XI, 2 (2011) 61–79
DOI: 10.2478/v10065-011-0010-y

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

Combining message encryption and authentication

Wojciech Oszywa*, Rafał Gliwa†

Military Communication Institute, 05-130 Zegrze, Poland

Abstract

The first part of the paper explains the need for combining message encryption and authentication. We begin with the example to emphasize the fact that privacy‡ does not imply authenticity. Then we prove, one needs both privacy and authenticity, even if one's aim is just getting privacy. In the second part we present an overview of different methods for providing authenticated encryption (AE) i.e. generic compositions, single-pass modes and two-pass combined modes. We analyze what are the advantages and disadvantages of different AE constructions. In the third part of the paper we focus on nonce§ based authenticated encryption modes. Our motivation is the wish to know the methodology of designing authenticated encryption mode of operation. We take into consideration a few most important properties, e.g. parallelizability, memory requirements and pre-processing capability. We analyze possibilities of choice of underlying encryption and authentication components and their order in a message we also try to answer. What does single-key mode really mean? Finally we mention the importance of provable security theory in the security of authenticated encryption modes.

*E-mail address: wojciech.oszywa@wat.edu.pl
†E-mail address: r.gliwa@wil.waw.pl

‡The terms 'privacy' and 'confidentiality' are used interchangeably, as defined in [**1**]

§Nonce (N) - a number used only once within a specified context.

1. The need for combining message encryption and authentication

We know we have to encrypt data to provide privacy, but quite often encryption is the way of providing the data authenticity, too. People assume that since decrypting with the wrong key will yield garbage, additional integrity checking is not needed. Unfortunately, it is not true. Suppose, for example, that the Sender S transmits an encrypted message M_{100} which indicates that the Receiver should please ask for transfer $100 from the checking account of S to the checking account of another party. The adversary A wants to change the amount of $100 to $900. She does not know the key K, so she cannot just encrypt M_{900} on her own. It seems that the privacy of C_{100} already rules out that C_{100} can be profitably tampered with. To see the flaws let us look at a counter-example [**2**]. If we encrypt M_{100} using a one time pad, then all the adversary has to do is to XOR the byte of the ciphertext C_{100} which encodes the character "1" with the XOR of the bytes which encode "1" and "9". That is when we one-time pad encrypt, the privacy of the transmission does not make it difficult for the adversary to tamper with ciphertext so as to produce related ciphertexts. The fact that data is encrypted need not prevent an adversary from being able to make the receiver recover data different from that which the sender had intended. Encrypting a message has never been an appropriate approach for protecting its authenticity. A good cryptographic design is goal-oriented. Message authentication oriented designs are Message Authentication Codes (MAC) algorithms [**1**].

But does one need authentication at all if one's aim is just getting privacy? Surprisingly, yes. There is a wealthy set of papers giving the evidence that one should never use encryption without providing authentication. Bellovin in [**3**] describes a number of attacks against various versions of IPSEC protocols, including confidentiality failures and authentication failures. One of them is cutting and pasting legitimate messages to decrypt someone else's traffic, caused by a lack of integrity checking. Borisov and Goldberg in [**4**] discovered several serious security flaws in the Wired Equivalent Privacy protocol used in 802.11 networks to protect data during wireless transmission. The WEP protocol uses an integrity checksum field to ensure that packets do not get modified in transit. The checksum is implemented as a CRC-32 checksum, which is part of the encrypted payload of the packet. The authors showed the importance of using a cryptographically secure Message Authentication Code to protect integrity of transmissions. The use of CRC checksum is completelly inappropriate for this purpose. CRC's are designed to detect random errors in the message, however, they are not resilient against malicious attacks. A secure MAC algorithm is particularly important in view of composition of protocols,

since the lack of message integrity in one layer of the system can lead to breach of secrecy in the larger system. Similar conclusions are presented by Vaudenay in [**5**] and Black and Urtubia in [**6**]. Vaudenay showed that symmetric encryption methods, which use padding, are vulnerable to side-channel weaknesses, when an adversary is able to distinguish between valid and invalid ciphertexts. Black and Urtubia demonstrated that such attacks are pervasive when the integrity is not guaranteed. If it were impossible for the adversary to produce valid ciphertexts other than those he has already seen, the mentioned attacks would vanish. The conclusion drawn from these examples is that encryption should always be accompanied by authentication. Then we are guaranteed that an adversary will not be able to take a given ciphertext and manipulate it to produce a new valid ciphertext. He will also not be able to combine two ciphertexts to produce a new valid ciphertext. In light of the described attacks, it is time to view authentication as a strongly-desirable property of any symmetric encryption scheme, including those, where privacy is the only security goal.

2. Authenticated encryption schemes

The term authenticated encryption scheme (AE scheme) is used to refer to a shared-key based transform, whose goal is to provide both privacy and authenticity of the encapsulated data [**7**]. By privacy we mean keeping information secret from all but those who are authorized to see it. By authenticity we mean corroborating the source of information (also known as data origin authentication), including data integrity i.e. ensuring information has not been altered by unauthorized or unknown means. In the authenticated encryption scheme the encryption process applied by the sender takes the key and a plaintext to return a ciphertext (including authentication tag), while the decryption process applied by the receiver takes the same key and a ciphertext to return either a plaintext or a special symbol indicating that it considers the ciphertext invalid or unauthentic. In many application settings we wish not only to encrypt and authenticate a message, but we wish also to include auxiliary data, which should be authenticated, but left unencrypted. An example might be a network packet where the payload should be encrypted and authenticated, but the header should be only authenticated (not encrypted). The reason is that routers must be able to read the headers of packets in order to know how to properly route them. This need caused some designers of AE schemes allow "associated data" to be included as input to their schemes. Such schemes have been termed authenticated encryption with associated data (AEAD) schemes (Fig. 1) [**8**].

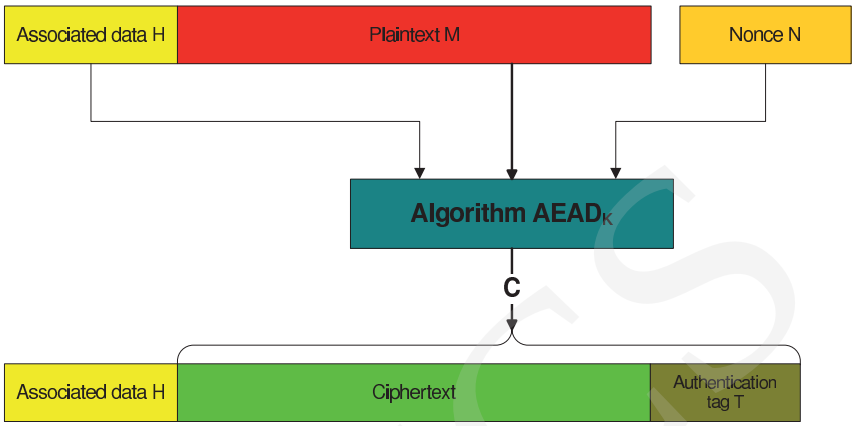

Fig. 1. AEAD general scheme.

2.1. Generic compositions

A traditional way of achieving both authenticity and privacy, called a generic composition [**7, 9**], was to simply find two algorithms yielding each of these properties and then use their combination on a message. It seems to be an obvious and completely safe approach. Unfortunately, such generic composition constructions are often ad hoc and it is very easy to accidentally combine secure encryption schemes with secure MACs and still get insecure authenticated encryption schemes [**7, 9**]. It is clear for researchers that one needs to use a keyed hash (i.e. MAC) with an appropriate key $K1$ along with a secure encryption scheme with an independent key $K2$. However, it is unclear in what order these modes should be applied to a message M in order to achieve authenticated encryption. There are three possibilities:

- MtE: MAC-then-Encrypt. We first MAC M under key $K1$ to yield tag T and then encrypt the resulting pair (M, T) under key $K2$;
- EtM: Encrypt-then-MAC. We first encrypt M under key $K2$ to yield ciphertext C and then compute $T = MAC_{K1}(C)$ to yield the pair (C, T);
- E&M: Encrypt-and-MAC. We first encrypt M under key $K2$ to yield ciphertext C and then compute $T = MAC_{K1}(M)$ to yield the pair (C, T).

The order of processing the message by the sender influences directly the order of decryption and verification processes for the receiver. In the case of MtE the message is first decrypted, then verified, in the case of EtM and E&M approaches it is first verified and only then decrypted. All these possibilities are used in practice by three popular protocols: SSL/TLS protocol uses MtE approach, IPSec uses EtM and SSH uses E&M. Do they all guarantee security?

Bellare and Nanprempre in [**7**] and Krawczyk in [**9**] showed, that EtM approach, with secure encryption scheme, secure authentication scheme and two independent keys is the best one for achieving AE. The security of remaining approaches i.e. MtE and E&M often depends on subtle details of how the data are encoded and on the particular MAC and encryption schemes used. The advantage of EtM paradigm is the possibility to discard a bogus message without necessity of decrypting it. But the disadvantage of EtM is that an adversary has access to the MAC's input data and to the MAC value, which makes him easier attack an authentication scheme. In MtE solution an adversary has access only to a ciphertext, so one can only try to attack encryption scheme. It is important, that in the case of MtE, as well as E&M, original plaintext is authenticated. Hence, there is no possibility of making mistake (as in the case of EtM), that after positive authentication verification by a receiver, he will use a wrong key for decryption and as a result, despite positive authentication, he will get a different plaintext from the sent one. A choice of the most proper solution is not clear and it is rather dependent on specific application requirements.

2.2. Single-pass modes

An alternative solution for providing authenticated encryption has become single-pass modes, which achieve simultaneously confidentiality and authenticity goals by processing every data block only once. The first correct single-pass mode was designed by Jutla, IACBC mode [**10**] (Fig. 2). A preliminary version of the mode just required a usual CBC encryption of the plaintext appended with the checksum, with a random initial vector r. But it turned out that such a scheme is susceptible to message integrity attacks. The solution was "whitening" the complete output with a random sequence i.e. XOR-ing the encrypted blocks with a random sequence. The random sequence could be generated by running the block cipher on $r+1, r+2, \ldots, r+t$, but with a different shared key. This requires m (the number of message blocks) additional cryptographic operations, and hence it is not more efficient than generating a MAC. The efficiency of the new mode comes from proving that the output whitening random sequence needs only be pairwise independent. In other words if the output whitening sequence is $s_1, s_2, \ldots, s_m$, then each s_i is required to be random, but only pairwise independent of the other entries. To get full independence, we must use the underlying block cipher, whereas to get pairwise independence it is sufficient to use combinatorial techniques. A simple algebraic construction in $GF(p)$ can generate such a sequence by performing only two cryptographic operations (two block cipher invocations).

In fact, an even weaker condition than pair-wise independence suffices, namely, the output whitening sequence s_i needs only be pair-wise differentially-uniform.

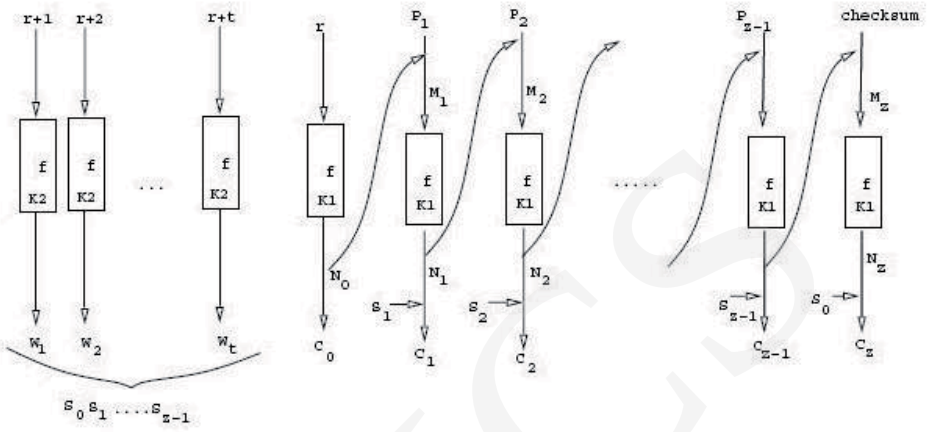

Fig. 2. IACBC scheme (source [**10**]).

A sequence of uniformly distributed n-bit random numbers $s_1, s_2, \ldots, s_m$, is called *pair-wise differentially-uniform* if for every n-bit constant c, and every pair i, j, $i \neq j$, probability that $s_i \oplus s_j$ is c is 2^{-n}. It is possible to generate such a sequence by performing only one cryptographic operation using a simple algebraic construction in $GF(p)$. The pair-wise differentially-uniform sequence s_i can also be used to remove chaining from the encryption mode while still assuring confidentiality. This results in a IAPM [**10**] mode of operation for authenticated encryption which is highly parallelizable. In both cases, IACBC and IAPM, checksum is expressed as XOR: $\sum_{i=1}^{z-1} P_i$, so the authenticity of a message is verified by checking if $P_z = P_1 \oplus P_2 \oplus \ldots \oplus P_{z-1}$. The cost of the encryption with authentication in IAPM is almost the same as the CBC or the CTR mode encryption only. A significant disadvantage of the IACBC/IAPM modes is the usage of two keys, instead of a single one. IACBC/IAPM do not have also an ability of processing headers that need to be only authenticated.

Gligor and Donescu have described the authenticated encryption modes XCBC and XECB [**11**], which are similar to Jutla's IACBC and IAPM, respectively. The main contribution of their work is the use of $\mod 2^n$ arithmetic for the calculation of the randomizing sequence (instead of $\mod p$ arithmetic as in the Jutla's modes).

Rogaway, Bellare and Black designed a single-pass scheme called OCB [**12**] (Offset CodeBook). This work was a follow-on to Jutla's IAPM scheme. In comparison to IAPM, OCB uses a single block-cipher key and does not require padding a message. The usage of a single key results from the fact that OCB uses special construction for calculating pairwise independent sequence that includes multiplication by a fixed element in the binary finite field $GF(2^n)$ and

Gray code calculations. OCB uses only $\lceil |M|/n \rceil + 2$ block cipher invocations (n - block length) to encrypt and authenticate a nonempty message M. OCB is parallelizable, thus it is suitable for encrypting messages in hardware at the highest network speeds (of the order 10 Gbps).

Unfortunately, all single-pass modes are patented. This has not been accepted by cryptographic community. That is why single-pass combined modes have never been used as a standard in any application. Furthermore, other researchers are afraid of continuing works in this field because they could be accused of violating some Intellectual Property. Hence, it was necessary to look for other, equally good but patent free solutions.

2.3. Two-pass combined modes

A lot of attention was paid to new block cipher modes after approval of new encryption standard AES by NIST in September 2000. Since that time a number of new constructions have appeared, including constructions, which achieve both privacy and authenticity. Besides single-pass modes described above, two-pass combined modes appeared. The first to be developed was the CCM, then EAX tried to solve some of the CCM problems, then CWC has been developed to improve EAX and finally GCM has been designed.

CCM [**13**] (Counter with CBC-MAC) is not much better than just generic composition because, in fact, it uses a standard MAC generation algorithm (CBC-MAC) and then a standard CTR encryption. The most important advantage of CCM is to use only one key to generate the MAC and encrypt. The two CCM processes are called generation-encryption and decryption-verification.

The input data to the generation-encryption process (Fig. 3) are a valid nonce N, a valid payload string M and a valid associated data string H (*header*), which are formatted according to the complex formatting function into block B. The CBC-MAC mechanism, using the initialization vector IV, is applied to the formatted data B to generate an authentication tag T. Then the counter mode encryption CTR is applied to the payload string M and separately to the tag T. The resulting data called the ciphertext, denoted C, is the output of the generation-encryption process.

The input to the CCM decryption-verification process is a purported ciphertext C (including encrypted authentication tag T), an associated data string H, and the nonce N that was used in the generation of the ciphertext. First, counter mode decryption is applied to the purported ciphertext to produce the corresponding tag T and payload M. If the nonce N, the associated data string H and the payload M are valid then these strings are formatted into blocks B according to the formatting function. Then the CBC-MAC mechanism is applied to verify the MAC. If verification succeeds, the decryption-verification

process returns the payload M as output. Otherwise, only the error message INVALID is returned.

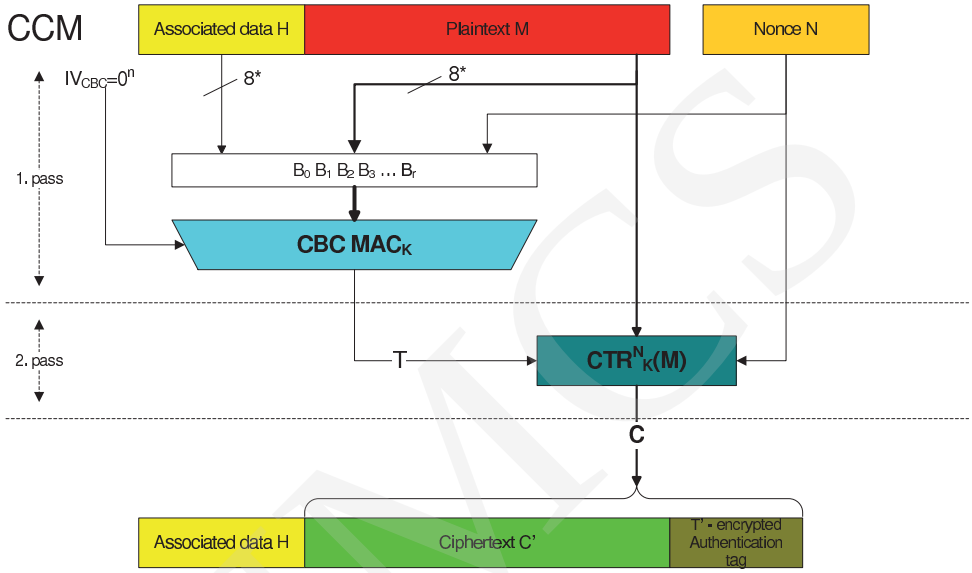

Fig. 3. CCM authenticated encryption scheme.

One of CCM disadvantages is a lack of ability to pre-process static associated data H (H that repeats in multiple messages should not affect the cost load of authentication after its first computation) as it encodes the nonce and the message length before the authentication. Another one is complex choice of parameters (e.g. message-length field size) and odd dependency between some of them (e.g. nonce length and the maximal message length). Unfortunately, CCM does not make on-line processing possible because it requires the length of the message to be known in advance. CCM uses a parallelizable counter mode for encryption but because of the use of CBC-MAC it can not be parallelized in general. A consequence of the usage of block cipher based CBC-MAC is a large number of block cipher calls: $2\lceil |M|/128 \rceil + \lceil |H|/128 \rceil + 2$. Nevertheless, CCM has been approved by NIST as the recommendation and specified in NIST Special Publications 800-38C. It has also become the mandatory mode for the 802.11 wireless networks.

EAX [**14**] solved some of the CCM problems, in particular the issue of knowing the message length in advance and its complicated definition that used some unnatural parameterization. EAX (Fig. 4) uses the CTR mode of operations, and then a modified CBC-MAC, called OMAC. EAX provides a nonce-based authenticated encryption with the associated data (AEAD) scheme. Given a single key K, a nonce N, a message M and a header H, EAX mode protects

the privacy of M and the authenticity of both M and H. The mode uses $2\lceil |M|/n \rceil + \lceil |H|/n \rceil + \lceil |N|/n \rceil$ block cipher calls (n - block length). EAX is on-line (the length of a message is not needed to begin its processing) and a static header can be pre-processed, reducing the cost of calculations.

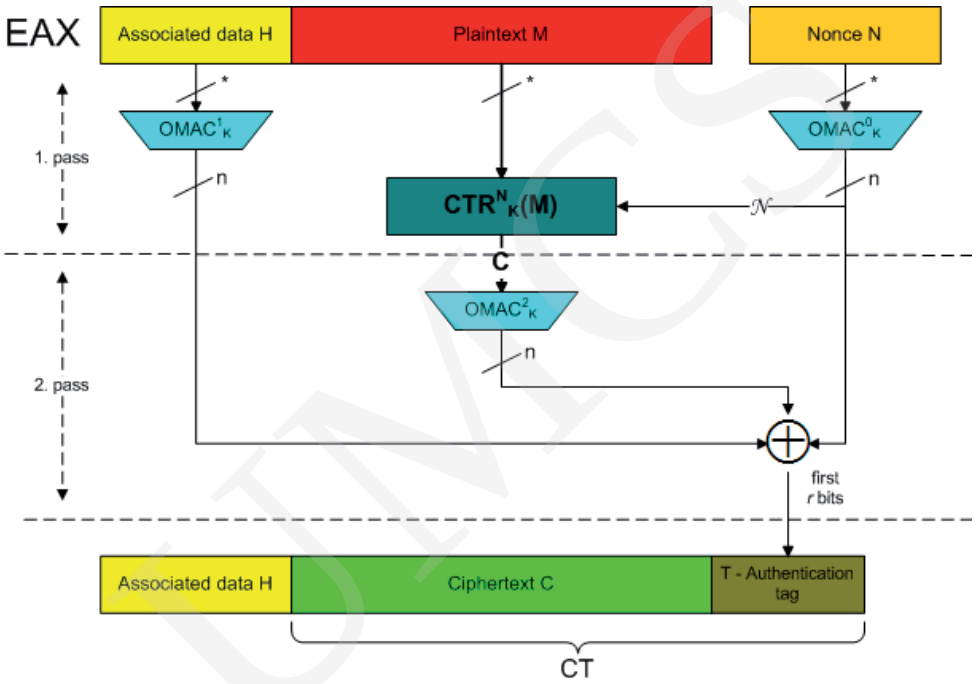

Fig. 4. EAX authenticated encryption scheme.

CWC [**15**] (CTR mode with the Carter-Wegman authentication) has been developed because both CCM and EAX can not be parallelized in hardware.

CWC (Fig. 5) also uses counter mode CTR for encryption, but the authentication tag is generated using the Carter-Wegman MAC construction. The Carter-Wegman hash function is evaluated as a polynomial modulo a prime number $(2^{127} - 1)$, which can be split into other polynomials and these compound polynomials can be processed in parallel. Then the hash value is encrypted. Instead of using multiple encryption operations in the MAC calculation, all one needs is to perform multiplication. CWC can run at 10 Gbps when using conventional ASIC technology and AES as the underlying block cipher (whereas CCM and EAX are limited to about 2 Gbps because of their serial constraints). Although CWC is based on the Encrypt-then-MAC paradigm, it is not a generic composition construction: the CWC encryption and MAC components share the same key. The total number of block cipher calls equals $\lceil |M|/128 \rceil + 3$.

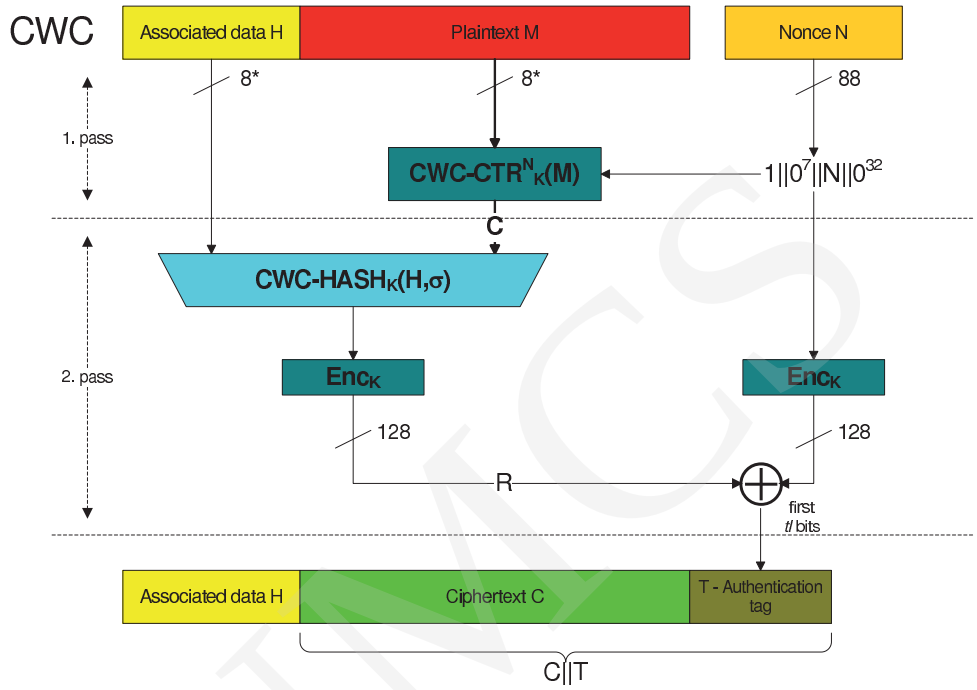

Fig. 5. CWC authenticated encryption scheme.

GCM [**16**] (Galois Counter mode) combines the well-known counter mode of encryption with the new Galois mode of authentication. It has been totally devoted to improve CWC performances and to fill its lacks. While CWC operated in a modulo 127 integer environment, the GCM is based on multiplication in the finite field with 2^{128} points. This allows GCM to perform with throughput of more than 10 Gbps in hardware. GCM (Fig. 6) encrypts first M in the counter mode CTR to get intermediate ciphertext C. Secondly, the associated data H is hashed together with C by the universal hash function GHASH. The resulting hash value is XOR-ed with the preprocessed and encrypted nonce N value to get the tag T. The final tag length is then fixed by the user's input most significant t bits. At the end, the chopped tag is appended to C to generate the final ciphertext. GCM construction allows the receiver of a bogus message to discard it before decryption, since the ciphertext, not the plaintext, is authenticated. GCM is the mode recommended by NIST in the NIST Special Publication 800-38D.

To present a complete survey of authenticated encryption schemes, it is necessary to mention also deterministic (nonce-less) authenticated encryption schemes (DAE) [**17, 18**] used to protect the transportation of cryptographic

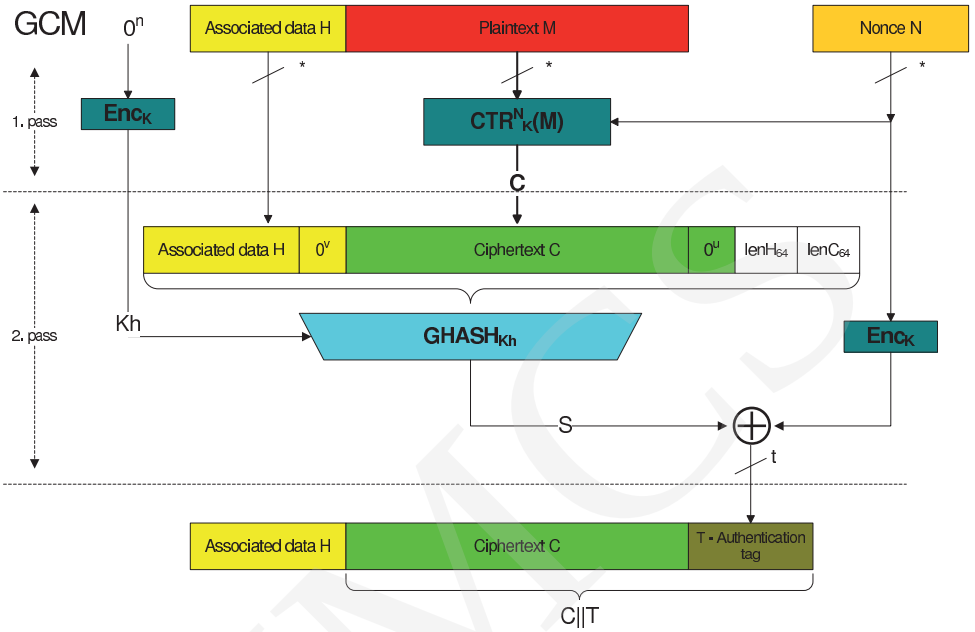

Fig. 6. GCM authenticated encryption scheme.

keys (key wrapping) and certain stream ciphers, which provide authentication by using a plaintext for the generation of a keystream [**19, 20**].

3. Consideration of AE mode designing

The National Institute of Standards and Technology (NIST) gives the requirements which a new block cipher mode submission should meet. From the designer's point of view, we are especially interested in the Summary of Properties item. It specifies main properties, which have to be taken into account by the designers of any block cipher mode of operation. These are:

- Security Function (encryption, authentication, authenticated encryption, hashing, pseudorandom bit generation, etc.),
- Error Propagation (e.g., none, m bits, n blocks, infinite),
- Synchronization,
- Parallelizability (e.g., sequential, interleaved, fully parallelizable),
- Keying Material Requirements (e.g., 1 key, 2 keys),
- Counter/IV/Nonce Requirements,
- Memory Requirements,
- Pre-processing Capability,
- Message Length Requirements (e.g., arbitrary length, padding necessary),

- Ciphertext Expansion (e.g., none, m bits, n blocks),
- Other Characteristics.

In relation to these properties, the designer of authenticated encryption mode should pay particular attention to the following problems.

Security Function. An authenticated encryption mode of operation must provide both privacy and authenticity of the message. The designer should consider also the possibility to include associated data, which should be authenticated but left unencrypted. This means that the encapsulation algorithm, on the input of a pair of associated data and message (A, M), and some nonce N, encapsulates (A, M) in a way that protects the privacy of M and the authenticity of both A and M.

Error Propagation (e.g. none, m bits, n blocks, infinite). Decryption of ciphertext containing bit errors may result in various effects on the recovered plaintext, including the propagation of errors to subsequent plaintext blocks. Different error characteristics are acceptable in various applications. Error propagation is the property that an error in the i-th ciphertext block is inherited by the i-th and all subsequent plaintext blocks. Modes with error propagation are well-suited for the situations where errors in transmissions are either unlikely to happen or taken care of by noncryptographic means like error-correcting codes, and situations, where an erroneous data transmission is dealt with by retransmission.

A special kind of error propagation is called error recovery. It is the property that an error in the i-th ciphertext is inherited by only a few plaintext blocks after which the mode resynchronizes. Modes with error recovery are suited for the situations where a retransmission after an erroneous data transmission is not possible or regarded too expensive.

In the case of authenticated encryption modes, error propagation and error recovery properties depend on the encryption component they use. But of course, the built-in authentication component causes, that all bit manipulations like flips and slips are detected and therefore no data are revealed.

Synchronization. Synchronization of authenticated encryption modes is based on a special number, used only once with a specific context, called nonce. The nonce shall be non-repeating in the sense that to any two distinct data pairs to be protected during the lifetime of the key distinct nonces should be assigned. The nonce must either be sent with the ciphertext or the receiver must know how to derive the nonce on its own.

Parallelizability. Parallelizability is significant advantage for hardware implementations. In order to enable parallel processing, one should consider using

encryption and authentication components of such construction, which will allow for parallelizability. Such components are for example the counter mode (CTR) for encryption and the Carter-Wegman scheme for authentication. Parallelizability requirement excludes the CBC-MAC type constructions and MAC schemes based on hash function because they work iteratively.

Keying Material Requirements. An authenticated encryption mode should operate single-key. The mode key is the same as the underlying block cipher key. In practice, AE modes do internally use two keys: a main block cipher key and a subkey for authentication, which is derived using the main block cipher key. Implementors can decide whether to store the derived authentication key in memory or whether to re-derive it as needed.

Counter/IV/Nonce Requirements. Authenticated encryption modes use a nonce, which is required to be a non-repeating value for a given key. To promote interoperability and simplicity of design it is recommended to fix the length of the nonce, for e.g. 96 bits. An AE scheme is considered secure as long as one does not query the encryption algorithm twice with the same nonce. The mode specification does not include nonce management, it remains the responsibility of users.

Memory Requirements. The memory requirements are basically those of the underlying block cipher. To minimize memory requirements it is worth providing privacy with a block cipher in such an encryption mode, for which it is enough to implement only a forward function of a block cipher algorithm. For example, in the counter mode the forward function is used for both encryption and decryption and an inverse cipher function is not used at all. Besides, using only one key gives additional savings.

Pre-processing Capability. Pre-processing capability significantly improves the efficiency of authenticated encryption. It is worth noticing that in the case of some encryption modes, the keystream can be pre-computed. Additionally, by giving the possibility of preprocessing associated data, one can reduce computation time if the associated data remains static or changes only infrequently.

Message Length Requirements. Authenticated encryption mode can process a message and associated data of any bit length or the bit length of the message and associated data may be limited to a multiple of 8 bits, i.e. each input string shall be an octet string. There does not appear to be a need to handle strings of arbitrary bit-length - applications adopt strings that are byte-aligned.

Ciphertext Expansion. The ciphertext expansion is the number of bits as ciphertext lengthened in relation to the plaintext, while providing a t-bit tag T. Depending on encryption component used, if the length of the last plaintext

block is smaller then multiple of the block size, padding may be necessary. To pad is to append a number of bits to that block, so that it becomes a multiple of the relevant block size. The desired state is, that on the input of a pair of an associated data and a message (A, M), mode outputs a ciphertext C with length $|C| = |M| + t$.

Other Characteristics. It is recommended to use a minimum number of options i.e. only the choice of the underlying block cipher and the tag length. Having fewer options makes interoperability easier.

It is also worth considering the possibility of processing data as it arrives, rather than waiting for the entire message to be buffered before beginning the encryption processes. This may be advantageous when encrypting streaming data sources. Note, however, that the decryptor should still buffer the entire message and check the tag before revealing the plaintext and associated data. To achieve on-line mode, the length of the messsage cannot be a necessary parameter to begin its processing.

What mostly affects the above mentioned parameters is the choice of underlying encryption and authentication components. Two of the most popular encryption modes are: Cipher Block Chaining (CBC) mode [**2**] and Counter (CTR) mode [**2**]. The CBC mode is a confidentiality mode whose encryption process features the combining ('chaining') of the plaintext blocks with the previous ciphertext blocks. The CTR mode is a confidentiality mode that features the application of the forward cipher function to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. Comparing the properties of CBC and CTR modes, the latter one seems to have more advantages with respect to the usage in the authenticated encryption mode. First, CTR uses only the forward cipher function in both encryption and decryption processes. It affects positively the size of hardware implementation. Second, the forward cipher functions can be performed parallel, so encryption as well as decryption can be performed parallel. In the CBC encryption, the input block to each forward cipher operation (except the first one) depends on the result of the previous forward cipher operation, so the forward cipher operations cannot be performed parallel. In the case of the CBC decryption the input blocks for the inverse cipher function, i.e. the ciphertext blocks, are immediately available, so that only multiple inverse cipher operations can be performed parallel. Third, the forward cipher functions in the CTR mode can be applied to the counters prior to the availability of the plaintext or ciphertext data. By contrast, in the CBC mode preprocessing is not possible because of a chaining mechanism.

Considering a choice of authentication component we have three alternatives [**21**]: Message Authentication Code (MAC) constructions based on block ciphers, MAC constructions based on hash functions and a universal hash-based MACs. MACs based on block ciphers are mostly based on the cipher block chaining mode. General problems with CBC-MACs are a lack of parallel processing (caused by the inherited CBC chaining dependence) and no possibility for preprocessing (the input block to each forward cipher operation depends on the result of the previous forward cipher operation). In the case of MACs based on hash functions, they form a MAC from hash functions by including a secret key as part of the hash input. Unfortunately such constructions (e.g. HMAC) also make parallel computations impossible, because they process the message iteratively. A compelling solution are universal hash-based MACs, introduced by Carter and Wegman [**22, 23**]. Instead of applying some cryptographic primitive directly to a message M to be MACed, the message is first hashed down to a smaller size using a hash function drawn from a Universal Hash Function Family, which had only a combinatorial property. Only then cryptographic primitive (e.g. one-time pad encryption) is applied to the smaller resulting string. The encrypted hash output serves as the MAC result. The universal hash-based MACs are more efficient than the previous approaches, because the universal hashing paradigm has reduced the efficiency problem of fast authentication to fast universal hashing. Moreover, the recent findings of internal collisions on some hash functions endanger the security of hash-based MACs. Thus, the universal hashing approach seems to remain at present the most perspective variants for the construction of secure and efficient MACs.

Having encryption and authentication components chosen, it is necessary to analyze consequences of a given order of processing a message. Should we first encrypt, then authenticate a message or first authenticate, then encrypt it. There are known theoretical results of Bellare [**7**], assuming some security notions that encryption before authentication is secure, but authentication before encryption not always and it often depends on subtle details of how the data are encoded and on the particular encryption and authentication components used. Ferguson and Schneier in [**23**] recommend not to be concerned abut these results because they do not apply to any ciphers in either CTR or CBC mode encryption.

Encryption before authentication gives the possibility of fast message verification by a receiver. The message is first verified and only when the verification was succeessful, decrypted. Such order of processing in decryption allows for an immediate discarding a bogus message without necessity of decrypting it, so

the verification is faster and the processor is less involved. But first encrypting, then authenticating a message results in the ciphertext which contains the input data for authentication component (i.e ciphertext) and the value of authentication tag available for an adversary. It gives the possibility of attacking the authentication component.

First authenticating, then encrypting a message (and authentication tag) causes, that both input data for the authentication component (i.e plaintext) and the value of authentication tag are hidden. Only the ciphertext with the encrypted authentication tag is available to an adversary, so attacking an authentication component is more difficult, one can only try to attack an encryption component.

Considering the choice of the processing order one has to take into account the Horton's principle "Authenticate what is being meant, not what is being said". Hence it seems, that authentication of ciphertext gives rise to a weak point. After positive message authentication verification, one can use a wrong key for decryption and as a result, despite positive authentication, he will get different plaintext from the sent one. In the case of plaintext authentication there is no risk of making such a mistake and we get assurance that after positive verification we get the original source plaintext. However, we mentioned earlier about the usage of a single key. In practice, authenticated encryption modes do internally use two keys: a main block cipher key and an authentication key, which is generated by applying the block cipher with the main key to the "zero" block. The main advantage of deriving the authentication subkey from the main key is that it saves memory storage space, simplifies key management and reduces the costs associated with fetching key material in hardware which can be a bottleneck.

An important parameter of security is the tag length. It must be fixed for any fixed value of the key. An attacker can attempt to forge a t-bit tag for a message by choosing it at random. In general, such an attack will succeed with the probability 2^{-t}. But there are also known attacks which increase probability of forgery, e.g. with GCM an adversary can choose tags that increase this probability, proportional to the total length of the ciphertext and associated data [**24**].

A crucial aspect of nonce-based AE modes is the correct usage of a nonce (a number used only once, within some established context). If there were no nonce there would be only one ciphertext for a given plaintext and key, and this means that the scheme would certainly leak information. A nonce is an input to the encryption process and the same nonce is required for the decryption operation. The nonce does not have to be random or secret or unpredictable.

A counter makes a perfectly good nonce, as does a random value. The nonce, however, needs to be handled with great care, because the misuse of nonce (i.e. repeating the same value) would generally lead to the complete collapse of system. It is the user's obligation to ensure that nonces do not repeat within a session.

With new objects it is often hard to know how much trust to put in their security. The problem with a protocol design is that a poorly designed protocol can be insecure even though the underlying atomic primitive is good. An example is ECB [**2**] (Electronic Code-Book) mode encryption with a block cipher. It is not a good encryption scheme because partial information about the plaintext leaks. Yet this is no fault of the underlying atomic primitive (e.g. AES). Rather, the atomic primitive was misused. In the past, to be sure of security of a new algorithm, it was necessary to give enough long time for good cryptanalysts to examine it. This approach was changing along with the introduction of the provable security theory. If the object is "primitive," such as a block cipher, no proof of security is possible, so instead we hope for security once we have shown that no known attacks (e.g. differential cryptanalysis) seem to work. However, for the algorithms which are built on top of these primitives, called "modes", we can prove some things about their security, namely that they are as secure as the primitives which underlie them [**25**].

4. Conclusions

As a short conclusion, let us quote the words of CWC mode designers [**15**]: "Combining encryption and authentication to get practical security is a difficult task. It is very easy to accidentally combine secure encryption scheme with secure MAC and still get insecure authenticated encryption scheme. This problem does not occur in the case of dedicated authenticated encryption modes because they clearly specify how to achieve both privacy and authenticity and there is no longer the risk of someone accidentally combining a privacy component with an authenticity component in an insecure way. Furthermore, since most applications that require privacy also require integrity, it is logical to focus on tools capable of providing both services simultaneously. There is thus great value in developing and standardizing dedicated AEAD schemes, as evidenced by a wealth of papers in this area."

References

[1] Menezes P., van Oorschot P., Vanstone S., Handbook of Applied Cryptography, CRC Press, New York (1997).

[2] NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques (December 2001).

[3] Bellovin S., Problem Areas for the IP Security Protocols, Proceedings of the Sixth USENIX Security Symposium (1996).
[4] Borisov N., Goldberg I., Wagner D., Intercepting Mobile Communications: The Insecurity of 802.11, ACM Press (2001).
[5] Vaudenay S., Security flaws induced by CBC padding - Applications to SSL, IPSEC, WTLS. Eurocrypt (2002).
[6] Black J., Urtubia H., Side-channel attacks on symmetric encryption schemes: The case for authenticated encryption, 11th USENIX Sec. Symposium (2002).
[7] Bellare M., Namprempre C., Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. Asiacrypt 2000, LNCS 1976, Springer-Verlag (2000).
[8] Rogaway P., Authenticated-encryption with associated-data. In ACM Conference on Computer and Communications Security, ACM Press (2002).
[9] Krawczyk H., The order of encryption and authentication for protecting communications (or: How secure is SSL?), In Advances in Cryptology - Crypto 2001, LNCS 2139, Springer-Verlag (2001).
[10] Jutla C. S., Encryption modes with almost free message integrity, Eurocrypt 2001, LNCS 2045, Springer-Verlag (2001).
[11] Gligor V., Donescu P., Fast encryption and authentication: XCBC encryption and XECB authentication modes, Fast Software Encryption 2001, 8th International Workshop, LNCS 2355, Springer-Verlag (2002).
[12] Rogaway P., Bellare M., Black J., OCB: A block-cipher mode of operation for efficient authenticated encryption. ACM Transactions on Information and System Security (TISSEC) 6, 3 (2003).
[13] NIST Special Publication 800-38C, Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality (May 2004).
[14] Bellare M., Rogaway P., Wagner D., The EAX mode of operation. FSE '04, LNCS 3017, Springer-Verlag (2004).
[15] Kohno T., Viega J., Whiting D., The CWC authenticated encryption (associated data) mode, http://eprint.iacr.org/2003/106/ (2003).
[16] NIST Special Publication 800-38D, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC (2007).
[17] Iwata T., Yasuda K., HBS: A Single-Key Mode of Operation for Deterministic Authenticated Encryption, LNCS 5665, Springer Berlin (2009).
[18] Rogaway P., Shrimpton T., The SIV Mode of Operation for Deterministic Authenticated-Encryption (Key Wrap) and Misuse-Resistant Nonce-Based Authenticated-Encryption (2007).
[19] Ferguson N., Whiting D., Schneier B., Kelsey J., Lucks S., Kohno T., Helix: Fast encryption and authentication in a single cryptographic primitive, In Fast Software Encryption, FSE 2003, LNCS 2887, Springer-Verlag (2003).
[20] Rose G., Hawkes P., Paddon M., Primitive specification for SOBER-128, available from http://www.qualcomm.com.au/Sober128.html (2004).
[21] Rompay B., Analysis and design of cryptographic hash functions, MAC algorithms and block ciphers. PhD Thesis, Catholic University Leuven (2004).
[22] Black J., Message Authentication Codes, PhD Thesis (2000).

[23] Carter L., Wegman M. N., New hash functions and their use in authentication and set equality, Journal of Computer and System Sciences 22 (1981).
[24] Ferguson N., Authentication Weaknesses in GCM, http://csrc.nist.gov/groups/ ST/toolkit/BCM/comments.html (2005).
[25] Bellare M., Practice-oriented provable-security, First International Workshop on Information Security, LNCS 1396, Springer-Verlag (1998).