# AUTHENTICATION AND CHOSEN CIPHERTEXT ATTACKS

BURTON ROSENBERG
UNIVERSITY OF MIAMI

## Contents



## 1. Chosen Ciphertext Attacks

Our development of ciphers resistant to attacks by adversaries of increasing access to the encryption process and decryption process, as given us not just better encryptions but concepts such as randomized encryption and malleability. Malleability, that a message can be manipulated through the encryption envelope, gives rise to Chosen Ciphertext Attacks (CCA) against encryption schemes that were otherwise Chosen Plaintext Attack (CPA) strong. The adversary queries for a decryption, then reliably modifies that ciphertext presented for decryption, and the resulting decrypted text, to give a new ciphertext–plaintext pair, that gives the adversary the advantage to with the indistinguishability game.

A necessary condition for CCA is then, that the encryption must not be *malleable*.

The theory at this point does something that is sufficient to insure malleability, it introduces the notation of *authenticity*, the ability to securely determine when a ciphertext was produced by the key holder. By securely, what is meant is that messages cannot be forged. Hence a manipulated ciphertext will be detected, and the malleability attack will fail.

In general, the logic runs (with perhaps a bit of hopefulness), if it is impossible to forge a ciphertext, the attacker can only proceed in a CCA with a genuine ciphertext obtained from a plaintext query, and hence has no need to ask for its decryption, as it already knows the plaintext. Indeed, a CPA strong encryption can be made

*Date*: September 23, 2023.

CCA if an authentication protocol is attached to it (in the proper way). This is called *authenticated encryption*. This strength is provably greater than simple CCA, however, just as Multiple Message is in practice achieved by having CPA, CCA in practice is achieved by having Authenticated Encryption.

## 2. Message Authentication Codes

A simple scheme for achieving authenticity is to choose a pseudorandom function, in fact a key $k$ for a pseudorandom function $F_k$, and for message $m$ calculate and sends $t = F_k(m)$ as the *Message Authentication Code (MAC)*, alongside the message $m$, as the pair,

$$\langle\, m, t \,\rangle, \quad t = F_k(m)$$

The receiver, on receiving a possibly corrupted or possibly forged $\langle\, m', t' \,\rangle$,and accepts only if $t' = F_k(m')$

For authenticated encryption, the encrypted version of the message is sent, not the message, but more about that later.

This protocol is more specific than need be. The verification method for a message, by which the MAC is recalculated and checked against an explicitly sent tag is perhaps not the only way verification can be done. It is called *canonical verification*. It seems also that the actual message $m$ got lost, replaced by an $m'$. This too is what we want, as we focus on any sort of forgery, not just a forgery dependent on additional constraints, as in, a meaningful forgery based on some desired effect.

**Definition 2.1** (**Existentially Unforgeable under an Adaptive Chosen Message Attack).** The PPT adversary $\mathcal{A}$ is given oracle access to the function generating MACs, and can ask for MAC's on any messages $m_i$. It then produces a pair $\langle\, m', t' \,\rangle$, such that $m' \neq m_i$ for any $i$.
The attack succeeds if the pair verifies as authentic. The MAC is secure if $\mathcal{A}$ succeeds with negligible probability, the probability over the space of keys and all randomness in the algorithms.

**Theorem 2.1.** If the tag is given by the pseudorandom function $t = F_k(m)$, and canonical verification is used, then the scheme is secure.

**Proof:** The usual distinguisher proof, where producing a forged tag will distinguish a pseudorandom function from a truly random function.

## 3. Authenticated Encryption

We can combine authentication and encryption to give a CCA strong cipher, which also has unforgeability, including non-mallebility. These are the sorts of encryptions used in practice. The full construction requires,

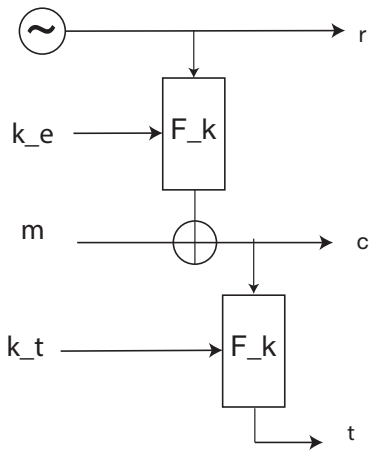


FIGURE 1. A CCA secure inencryption, for fixed block length

(1) Two keys, one for encryption the other for authentication,
(2) Randomized encryption,
(3) A protocol that does not leak information through manipulation of the MAC/verification component.

**Definition 3.1** (**A CCA and authenticated encryption).** Chose two keys $k_e$ and $k_t$. For message $m$ choose a random $r$ and send,

$$\mathcal{E}_{k_e,k_t} \mapsto \langle\, r, c, t \,\rangle, \quad c = F_{k_e}(r) \oplus m,\ t = F_{k_t}(c)$$

There are subtle ways in which combining a MAC can affect security. MAC's that are deterministic, as most are, must be diversified in their inputs, else the MAC leaks when two encryptions are of the same message. All of these schemes set aside ciphertexts that are legal from those that are not, and those that are legal should be sufficiently rare. It might be possible to use the legal versus illegal as an oracle to learn things about the plaintext. This has been exploited both on paper and in the real world with padding attacks.

The proposed scheme MAC's a publically provided value, the $c$. Hence any attack to this scheme can be an existential forgery attack.

## 4. Separation Proofs for the security hierarchy

**Theorem 4.1.** There is an encryption scheme is eavesdropping secure which is not Multiple Message Attack secure.

**Proof:** In fact, an encryption which is deterministic cannot be multiple message attack secure. Consider the on time pad with key $k$. The encryption is $\mathcal{E}_k(m) =$

$m \oplus k$. Hence the adversary can give these two message vectors, $M_0 = \langle m_1, m_1 \rangle$ and $M_1 = \langle m_1, m_2 \rangle$, with $m_1 \neq m_2$. If $b = 0$ then the returned challenge is $\langle c, c \rangle$, but if $b = 1$ the returned challenge is $\langle c, c' \rangle$ with $c \neq c'$. These are distinguished with probably 1. □

**Theorem 4.2.** There is an encryption scheme with is Multiple Message secure which is not CPA (Chosen Plaintext Attack) secure.

**Proof:** Actually, I am not going to prove this, just comment on where to find a proof. The difference between multiple message and CPA is CPA is adaptive. Therefore, a successful CPA attack against a Multiple Message secure scheme must use a result of an encryption in a future encryption query. I do not yet know an example that does not feel contrived.
*Notes:* The multiple message for CPA goes back to the CPA definition with a modification, resulting in LR-CPA. Then there is the burden to show that CPA and LR-CPA have identical security properties. The difference between CPA and LR-CPA is $b$ is involved only once in CPA but in all queries in LR-CPA. However, the original Multiple Message attack is of the LR format.

**Theorem 4.3.** Any LR-CPA (Left-Right CPA) secure encryption is CPA secure.

**Proof:** This assures our hierarchy is properly drawn. Suppose that the encryption is not CPA secure. The adversary $\mathcal{A}$ in the CPA game gives an adversary $\mathcal{A}'$ in the LR-CPA game. Specifically, when $\mathcal{A}$ asks to encrypt $m$, the $\mathcal{A}'$ asks to encrypt $\langle m, m \rangle$. What ever the hidden $b$, the result is $\mathcal{E}_k(m)$, and the $\mathcal{A}$ attack can continue. When $\mathcal{A}$ produces its query $M = \langle m_0, m_1 \rangle$, so does $\mathcal{A}'$ and receives $\mathcal{E}_k(m_b)$, and the $\mathcal{A}$ attack can continue. □

**Theorem 4.4.** Any CPA secure encryption is LR-CPA secure.

**Proof:** I shall only sketch this. Consider a LR-CPA attack where the oracle interactions are,

$$\langle m_{i,0}, m_{i,1} \rangle_{i=0,1,\ldots,t} \mapsto \langle \mathcal{E}_k(m_{i,b}) \rangle_{i=0,1,\ldots,t}.$$

Consider a modification of this interaction, parametrized on $j$,

$$\langle m_{i,0}, m_{i,1} \rangle_{i=0,1,\ldots,t} \mapsto \langle \mathcal{E}_k(m_{i,b(i)}) \rangle_{i=0,1,\ldots,t}.$$

where

$$b_j(i) = \begin{cases} 0 & i < j \\ b & i = j \\ 1 & j < i \end{cases}$$

This sequence is the true LR experience when $j = 0$ or $j = n$, depending on $b$. Each step of the way, however it is a CPA attack, not an LR-CPA attack.

If between the extremes for $j$, overall, there is a non-negligible advantage, at some step there has to be a non-negligible advantage, as there are only a polynomial number of steps. For that $j$, this implies a successful CPA attack. □

**Theorem 4.5.** There is an encryption scheme with is CPA secure which is not CCA (Chosen Ciphertext Attack) secure.

**Proof:** Consider the CPA secure encryption $\mathcal{E}_k(m) = \langle r, m \oplus F_k(r) \rangle$, where $r$ is a randomly chosen $n$ bit string. Let

$$c = \mathcal{E}_k(m_b) = \langle r, m_b \oplus F_k(r) \rangle$$

be the challenge. The adversary asks for the decryption

$$\mathcal{D}_k(\langle r, 0 \rangle) = 0 \oplus F_k(r) = F_k(r)$$

From this the adversary recovers $m_b$. We assume that $c \neq \langle r, 0 \rangle$. This will be so with overwhelming probability. □

**Theorem 4.6.** If a CPA secure scheme has encryptions signed by a strongly unforgeable MAC, the result is CCA secure.

**Proof:** Consider an CCA adversary. Any decryption query must be properly authenticated. This means that a pair $\langle c, t \rangle$ is presented that verifies. Such a query is the result of a previous encryption query, or the adversary has successfully forged a signature on $c$. This last case is only of negligible probability.

Hence a CCA attack will only use the decryption oracle for previous encrypted data, except for a negligible number of instances, and hence will be a CPA attack. □

## 5. Some other related proofs

The CCA secure construction presented requires a strong unforgeable MAC. We have a one block MAC given a pseudo-random function. A MAC for multiple blocks can be constructed as a CBC-MAC. However, it is unforgeable only if the message space does not include any two messages, one a prefix of the other. If this is not ture, the following forgery is possible, called a length extension attack.

**Theorem 5.1.** It is possible to forge a CBC-MAC when the block length is placed as the last block.

**Proof:** The adversary queries for the tags $A$ and $B$ one the block messages $a$ and $b$,

$$A = E_k(1 \oplus E_k(a)), \quad B = E_k(1 \oplus E_k(b))$$

The adversary queries for the tag $D$ on the three block message $a||1||A$,

$$D = E_k(3 \oplus E_k(A \oplus E_k(1 \oplus E_k(a)))) = E_k(3 \oplus E_k(A \oplus A)) = E_k(3 \oplus E_k(0)).$$

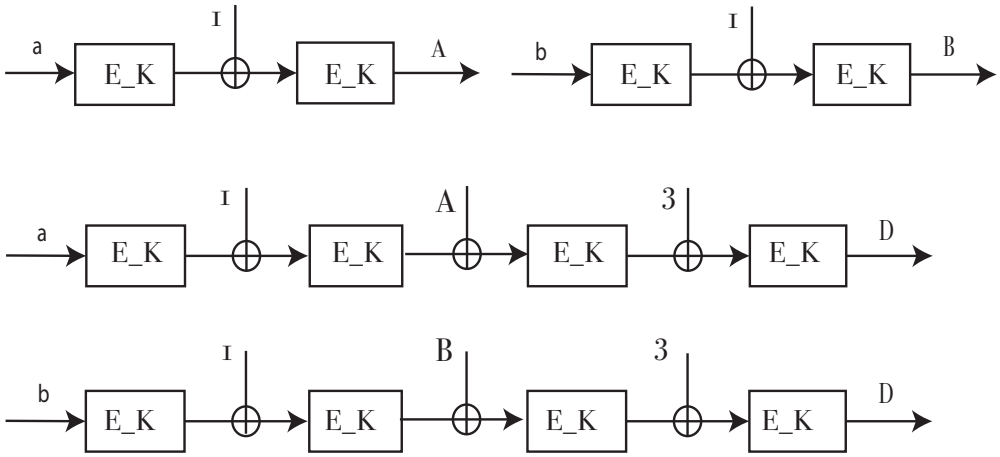


FIGURE 2. Forgery Length Extension.

The successful forgery is that the tag on $b||1||B$ is also $D$,

$$E_k(3 \oplus E_k(B \oplus E_k(1 \oplus E_k(b)))) = E_k(3 \oplus E_k(B \oplus B)) = E_k(3 \oplus E_k(0)) = D.$$

See Figure 2. □

Another way to construct a MAC is the Hash-then-MAC scheme, which uses the Merkle-Damgard construction to domain extend a collision-resistant hash from a fixed length to a variable length. We wish to note here now careful one must be to have collision-resistance of the construction. For instance, control over the IV will give a simple process to find a collision.

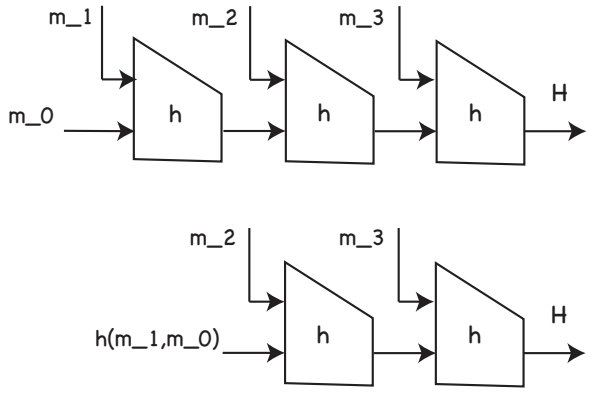


FIGURE 3. Wrong example of Merkle Damgard