1 Authentication

Authentication is the process of determining whether someone or something is, in fact, who or what it claims to be.

![Figure 1: A sends a message M to B.](image)

Here we see that a message M was sent by A and received by B, but what is the guarantee that the message was in fact really sent by A and not by any other adversary pretending to be A? In order to verify this we need to use a mechanism that will help us verify the identity of the sender of the message so that we can establish authenticity.

To achieve Authentication we need to use some type of a mechanism that binds the message with the identity of the sender. Let us consider the case of symmetric key encryption. Authentication is achieved as shown,

![Figure 2: A sends message M along with an authenticator to B.](image)

Here Auth_k(M) is the Authenticator for A, which is nothing but the message M encrypted with Secret key K. K is unique and is shared only by the communicating parties in a Symmetric key encryption scheme.
Now, the question can be whether confidentiality (achieved by encryption) imply Authentication, in general? The answer to this question is actually “No”. We cannot guarantee authentication in such a situation, since, in the above method, the key K may be revealed, which could allow the adversary to create forgeries. To understand this better let us look at the case of One Time Padding (OTP). In conclusion, any encryption scheme cannot be used to achieve authentication. Authentication and Confidentiality are two different requirements and they both have different properties and goals.

Figure 3: Adversary E sends message M' with a modified authenticator to B.

2 Message Authentication Code(MAC)

**Definition:** Usage of a message authentication code procedure involves an algorithm that accepts a secret key and a message as input; and produces a MAC as output. This process provides both an integrity check (by ensuring that a different MAC will result if the message has been altered) and an authenticity check (because only the person knowing the secret key could have produced a MAC).

**KeyGen:** $k \leftarrow K$

**MAC:** Input: $m, k$
$\mu \leftarrow MAC(m, k) = f_k(m)$.
Output: $(m, \mu)$

**Verify:** Input: $(m, \mu)$ and $k$
$Yes/No \leftarrow \text{Verify}(m, \mu, k)$
Thus, we achieve authentication of the sender using MAC.

3 Security Notions

An adversary can successfully thwart the MAC system if he forges a message $M'$ such that it passes the verification. To avoid such a situation, MAC needs to be secure in terms of a set of security notions. They are as follows

3.1 Security against Key Recovery

Securiy against key recovery means, given several messages and MAC pairs $(m_i, \mu_i)$, it should not be possible to learn the key $K$. It is easy to see that security against key recovery is a necessary condition, it is not a sufficient (or the strongest) condition that needs to be satisfied. There are several ways by which an adversary can forge the messages without having to recover the key. Also, we may not need the key to forge a message.

3.2 CMA Security: Existential Forgery against adaptively chosen message attacks

Here, we consider a scenario where we have a signing oracle and verification oracle. Signing oracle is responsible for producing the MAC output which is $\mu_i$ when given an input of $m_i$, using the key $k$. Verification oracle is responsible for checking that the received MAC is valid for that corresponding message.

The adversary $A$ queries the signing oracle with various messages and is returned with the corresponding $\mu$ that is calculated. The main aim of the adversary here is to forge a message $m$ and a possible authenticator for that message so as to pass the verification process. The forged message here is different from the messages that $A$ had queried earlier to the signing oracle.

The adversary $A$ will now query the verification oracle with the arguments $(m, \mu)$ and as a result gets to find out if the forging was successful or not.

If $\forall A$ who are playing such a game, if the advantages are negligible we say that MAC scheme is CMA secure.

In the above authentication protocol, we see that $A$ acts as a signing oracle for the adversary and $B$ as a verification oracle. It is expected that adversary will come up with a forged message for the challenge $R$ that $B$ had given to $A$. But here we restrict the forgery of the message to a limited space. When defining CMA we do not restrict the content of the forged message, i.e., the forged message need not be an exact suitable reply
Figure 4: CMA security with the Adversary attempting to forge a message.

Figure 5: The Adversary sends the forged message to B.
to a challenge that the other party had given. Thus this makes it an Existential CMA and also shows that it is more secure. In other words, any scheme that is secure against existential forgery will also be secure in the context where the adversary needs to come up with a forgery on a message not of his/her choice.

4 MAC Construction

There several ways to construct a MAC, each have its advantages and disadvantages.

4.1 A PRF is a CMA-secure MAC

\[\text{Figure 6: PRF used on message of length } l \text{ to generate a MAC of length } l.\]

A PRF, using a key \(k\), can be used as a (deterministic) MAC using following steps:

- **KeyGen**: \(k \leftarrow K\)
- **MAC**: \(\mu = MAC_k(m) = f_k(m)\). \((m, \mu)\) is sent to the recipient.
- **Verify**: Return \(f_k(m) == \mu\)

**Theorem 1 (f-PRF \(\Rightarrow\) MAC\(_f\)-CMA)** If \(f\) is a PRF, then MAC scheme using \(f\) is CMA-secure.

**Proof.** We will prove the contrapositive, \(\neg\text{MAC}_f\text{-CMA} \Rightarrow \neg\text{f-PRF}\). See Figure 7.

\(\mathcal{A}\) thinks that it is interacting with Signing and Verification oracles of the MAC scheme, whereas, it is infact it is communicating with \(\mathcal{B}\). It sends message \(m_i\) to \(\mathcal{B}\). \(\mathcal{B}\) sends \(m_i\)
to the challenger. The challenger chooses a world, and returns $MAC_i = g(m_i)$, where $g()$ depends on the world the Challenger chooses. $\mathcal{B}$ sends $MAC_i$ to $\mathcal{A}$ (this way $\mathcal{B}$ simulates the signing oracle to $\mathcal{A}$). $\mathcal{A}$ produces a possible forgery: a message $M$, and its corresponding message authentication code $MAC'_\text{new}$, and sends it to $\mathcal{B}$. $\mathcal{B}$ sends message $M$ to the challenger. The challenger takes $M$, and return $MAC' = g(M)$. $\mathcal{B}$ compares if $MAC'_\text{new}$ equals to $MAC'$. If they are equal, then it sends $d = 1$ to $\mathcal{A}$ (this simulates the verification oracle to $\mathcal{A}$) and the challenger, otherwise it sends $d = 0$ to $\mathcal{A}$ and the challenger.

$$\text{Adv}_{f}^{\text{PRF}}(\mathcal{B}) = Pr(\text{Exp}_{f}^{\text{PRF}^{-1}}(\mathcal{B}) = 1) - Pr(\text{Exp}_{f}^{\text{PRF}^{-0}}(\mathcal{B}) = 1)$$

$$= \text{Adv}_{f}^{\text{CMA}}(\mathcal{A}) - \frac{1}{2^l}$$

$$\Rightarrow \text{Adv}_{f}^{\text{PRF-MAC}}(\mathcal{A}) = \text{Adv}_{f}^{\text{PRF}}(\mathcal{B}) + \frac{1}{2^l} \leq \epsilon + \frac{1}{2^l}$$

This completes the proof.

**Note:** While this construction of MAC using a PRF is simple, it doesn’t allow authentication of messages larger than $l$ bits.
4.2 How do we authenticate arbitrarily long messages?

We use chaining mechanism to create MACs on arbitrarily long messages. This cascading effect securely guards against 2 similar blocks of messages. (Refer to figure 8)

\[ \mu_i = F_k (\mu_i) \]
\[ (m_1, m_2, ..., m_n, \mu_1, \mu_2, ..., \mu_n) \]

In this case, authenticator is independent of the size of the message.

However, this method is not secure because we can scramble the message:

\( (m_1, m_2, ..., m_n, \mu_1, \mu_2, ..., \mu_n) \) can be changed to \( (m_2, m_1, ..., m_n, \mu_2, \mu_1, ..., \mu_n) \)
4.3 CBC-MAC

Longer messages can be encoded by dividing a message into blocks and chaining them, similar to CBC-DES. Refer to figure 9.

This chaining is different than CBC-DES; there is no IV value (and therefore the MAC is deterministic) and also the output is only one block.