CSC 591 Cryptography 10/23/2018

Lecture : Message Authentication Codes

Lecturer: Alessandra Scafuro *Scribe: Dhruvil Mehta*

Topic

We know that confidentiality is a necessary condition for secure communication. Private communication necessitates that the message being sent over the communication channel should not be made available to an adversary on the channel.

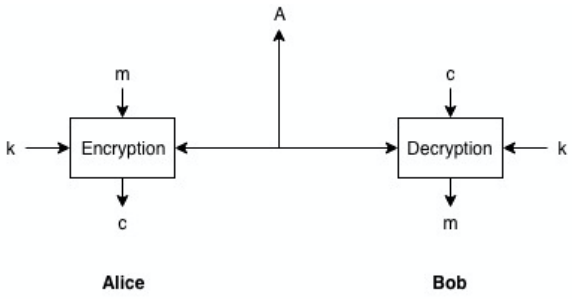

Figure 1: Communication with the presence of an Adversary

In the above figure, communication between Alice and Bob can be considered as private if and only if the adversary does not get access to the message being sent, or access to additional information which can be gleaned off of the ciphertext. But confidentiality is not the only aspect of security. Another important aspect is **Message Authentication**.

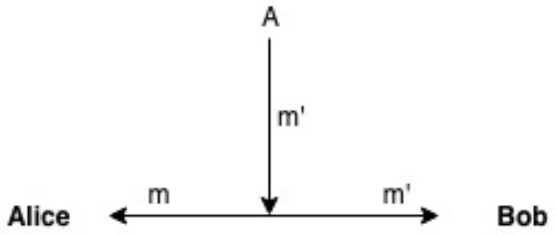

Figure 2: Adversary in Authentication

The message authentication problem can be summarized by the figure above: Alice sends a message m meant for Bob over the communication channel. An adversary intercepts the message, and replaces it with another message m' of the adversarys choice. Bob now faces a predicament. Did Alice really mean m'? In other words, is m' an authentic message?

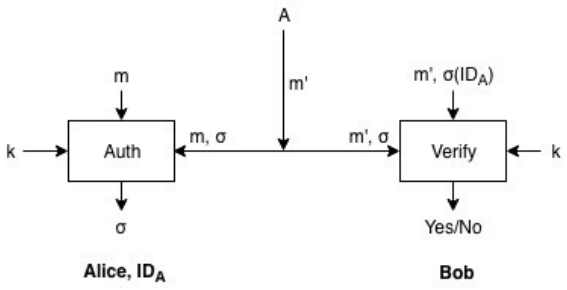

Figure 3: Solution for Authentication

The figure above proposes a solution to the authentication problem: Alice generates an authentication code σ (aka tag) which is based on the message and a key and sends the message as well as the code to Bob. The adversary replaces m with m', but cannot generate an authentication code σ for m'. When the message tuple reaches Bob, he verifies whether σ corresponds to the message m'. Since it does not, Bob rejects the message as unauthentic. Such authentication codes are known as **Message Authentication Codes**.

Syntax of MAC: A message authentication code (or MAC) consists of three probabilistic polynomial-time algorithms (Gen, Mac, Vrfy) such that:

1. The key-generation algorithm Gen takes as input the security parameter 1^n and outputs a key k with $|k| \leq n$.

2. The tag-generation algorithm Mac takes as input a key k and a message $m \in \{0,1\}^*$, and outputs a tag t. Since this algorithm may be randomized, we write this as $t \leftarrow \text{MAC}_k(m)$.

3. The deterministic verification algorithm Vrfy takes as input a key k, a message m, and a tag t. It outputs a bit b, with $b = 1$ meaning valid and $b = 0$ meaning invalid. We write this as $b := \text{Vrfy}_k(m, t)$.

A MAC involves the use of a secret key for both the generation of the tag and the verification of the tag. To provide a system for authentication in the public setting, **Digital Signatures** are used. There are major differences in the two implementations:

- A MAC algorithm requires that users agree upon the same secret key such that tag-generation and verification is possible.
  On the other hand, digital signatures make use of the private key of the sender to sign a message, and the public key of the sender to verify it. The private key remains secret, that is, it is not shared with anyone.

- Due to the presence of a shared secret, MAC cannot provide the property of non-repudiation, since any individual who is capable of verifying a MAC is also capable of generating a MAC for another message.
  Digital signatures provide the property of non-repudiation since they work on the principle of public key cryptography.

Definition

If an adversary attempts to attack the message authentication code scheme, he must be able to forge the MAC of the corresponding message, as seen in figure 4. Therefore, in order to define the security of message authentication codes, we need an experiment that requires the concept of Unforgeability,

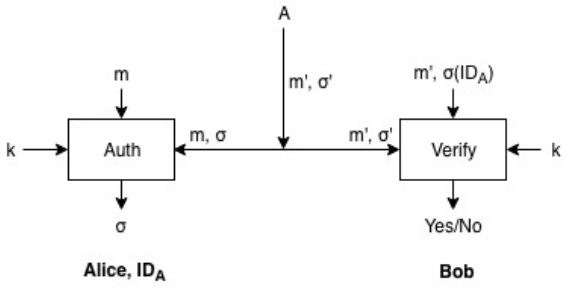

Figure 4: Adversary Forging σ' Successfully

To satisfy the condition of Unforgeability, the situation described in the above figure should not occur, i.e., the adversary should be unable to forge a tag σ corresponding to m' which can be independently verified as the authentication code by Bob as that of m'

Towards a formal definition for a MAC algorithm, consider the following experiment for a MAC Π = (Gen, Mac, Vrfy), and adversary A and value n as the security parameter.

Definition of the message authentication experiment Mac-forge$_{A,\Pi}(n)$:

1. A key k is generated by running Gen(1^n).
2. The adversary A is given the input 1^{n} and oracle access to MAC$_k(\cdot)$. A is allowed to query the oracle polynomial number of times. The adversary eventually outputs a forgery (m^*, t^*). Let Q denote the set of all queries that A asked its oracle.
3. A succeeds if and only if:
   - Vrfy$(m^*, t^*) = 1$ and
   - $m^* \notin Q$

In that case, the output of the experiment is defined to be 1.

A MAC is deemed secure if there exists no PPT adversary who can succeed in the above experiment with non-negligible property.

Unforgeability of MAC: A message authentication code Π = (Gen,Mac,Vrfy) is strongly secure, or a strong MAC, if for all probabilistic polynomial-time adversaries A, there is a negligible function negl such that

$$Pr[\text{Mac-forge}_{A,\Pi}(n) = 1] \leq \mathsf{negl}(n)$$

Scheme $\Pi_{\mathbf{MAC}}$

Key Generation Algorithm $\mathrm{Gen}(1^n)$: randomly sample $k \leftarrow \{0,1\}^n$.
Tag Generation Algorithm $\mathrm{Mac}(m,k)$: Compute $t = F_k(m)$ and output the tag t. (If $|m| \neq |k|$ then output nothing.)
Verification Algorithm $\mathrm{Vrfy}(m,k,t')$: Check whether $t' = F_k(m)$, if yes then output 1; otherwise output 0. (If $|m| \neq |k|$ then output 0)

Theorem

Assuming that F_k is a PRF, then Π_{MAC} is a secure (fixed-length) MAC scheme.

Security Proof

Towards a contradiction, assume PPT adversary $\mathrm{A}_{\mathrm{forge}}$ that succeeds in the message authentication experiment with non-negligible probability $p(n)$.
Namely, $\mathrm{A}_{\mathrm{forge}}$ outputs a forgery (m^*, t^*) such that $t^* = F_k(m)$ with probability $p(n)$.

Then we can construct an adversary of the PRF $\mathrm{A_F}$ that distinguishes the output of PRF F_k from a truly random function (TRF).

Reduction:

1. $\mathrm{A_F}$ has oracle access to a function $\mathcal{O}(\cdot)$. $\mathcal{O}(\cdot)$ can be either a pseudorandom function PRF or a truly random function TRF.
2. When $\mathrm{A}_{\mathrm{forge}}$ queries with message m_i, $\mathrm{A_F}$ forwards m_i to the oracle $\mathcal{O}(\cdot)$ and receives $y_i \leftarrow \mathcal{O}(m_i)$.
3. When $\mathrm{A}_{\mathrm{forge}}$ outputs the forgery (m^*, t^*), then $\mathrm{A_F}$ sends m^* to the oracle to obtain $z \leftarrow \mathcal{O}(m^*)$.
4. If $z = t^*$ and $\mathrm{A}_{\mathrm{forge}}$ never queried its MAC oracle on m^*, $\mathrm{A_F}$ outputs 1, otherwise $\mathrm{A_F}$ outputs 0.

Notice that if $\mathrm{A_F}$'s oracle is a pseudorandom function, then the view of $\mathrm{A}_{\mathrm{forge}}$, which is run by $\mathrm{A_F}$ as a sub-routine is distributed identically to the view of $\mathrm{A}_{\mathrm{forge}}$ in experiment $\text{Mac-forge}_{A_{\mathrm{forge}},\Pi}(n)$. Namely

$$\Pr[{A_{\mathrm{F}}}^{F_k(\cdot)}(1^n) = 1] = \Pr[\text{Mac-forge}_{A_{\mathrm{forge}},\Pi}(n) = 1]$$

where k is chosen uniformly at random.

We define the experiment $\text{Mac-forge}_{A,\tilde{\Pi}}(n)$ is the same as the experiment $\text{Mac-forge}_{A,\Pi}(n)$ except that a truly random function $\mathrm{TRF}(\cdot)$ is used instead of the pseudorandom function in the scheme above.

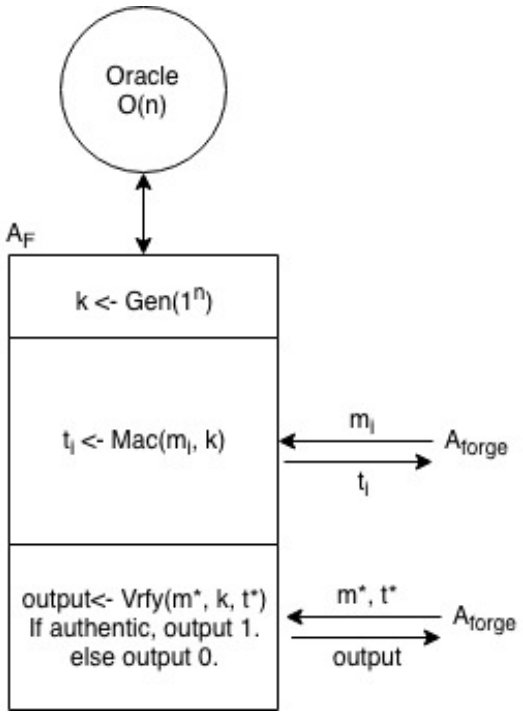

Figure 5: Reduction in effect

If $\mathrm{A_F}$'s oracle is a random function, then the view of $\mathrm{A_{forge}}$, which is run by $\mathrm{A_F}$ as a subroutine is distributed identically to the view of $\mathrm{A_{forge}}$ in experiment Mac-forge$_{A_{\mathrm{forge}},\tilde{\Pi}}(n)$. Namely

$$\Pr[{A_{\mathrm{F}}}^{\mathrm{TRF}(\cdot)}(1^n) = 1] = \Pr[\text{Mac-forge}_{A_{\mathrm{forge}},\tilde{\Pi}}(n) = 1]$$

where TRF is a truly random function.

Therefore we have

$$|\Pr[{A_{\mathrm{F}}}^{F_k(\cdot)}(1^n) = 1] - \Pr[{A_{\mathrm{F}}}^{\mathrm{TRF}(\cdot)}(1^n) = 1]| = |p(n) - \frac{1}{2^n}|$$

which is non-negligible. This contradicts to our assumption that F_k is a secure PRF.

References

[1] J. Katz, Introduction to modern cryptography. Boca Raton, FL: Chapman & Hall/CRC, 2008.