Cryptography
Lecture 10

Lecture by Dr. Alon Rosen
Typeset by Steven Karas

2019-05-28
Last edited 18:13:49 2019-05-28

Disclaimer These notes are based on the lectures for the course Cryptography, taught by Dr. Alon Rosen at IDC Herzliyah in the spring semester of 2018/2019. Sections may be based on the lecture slides prepared by Dr. Alon Rosen.

1 Recap

I was not present in the last three lectures due to scheduling conflicts.

2 Agenda

- MAC

3 Message Authentication (MAC)

Note that encryption in and of itself does not provide authentication. The rough idea is to attach a "tag" to the message to prove that the message is authentic. MAC provides the symmetric encryption equivalent of asymmetric signatures. Note that asymmetric signatures can be verified by anyone, whereas MACs can only be authenticated by those who know the key.

For example if Alice wants to send a message to Bob where they both have a shared key k, Alice can send $(m,t)_k$ to Bob who can verify that the message tag corresponds to the key k. The security of this authentication is limited by knowledge of the key.

3.1 Definition

A MAC consists of three algorithms (G, M, V):

1. A key generation algorithm $k \xleftarrow{R} G(1^n)$
2. A tagging algorithm $t \xleftarrow{R} M_k(m)$
3. A verification algorithm $V_k(m,t) \in \{\text{ACC}, \text{REJ}\}$

$$\forall m \; V_k(m, M_k(m)) = \text{ACC}$$

Adversarial Model Given an adversary F, their goal is to produce a valid tag for message they have never seen a valid tag for. Assume that F can request from an oracle valid tags $t_i = M_k(m_i)$ for any number of messages m_i. We say that F forges $V_k(m, M_k(m)) = \text{ACC}$ for some $m \neq m_i \forall i$.

3.2 Existential unforgeability under adaptive chosen message attack

A MAC (G, M, V) is secure if for any PPT F:

$$\exists \, \text{neg}\, \varepsilon : \Pr\left[F^{M_k(\cdot)}(1^n) \text{ forges}\right] \leq \varepsilon$$

3.3 Fixed length MAC

$M_k(m) = f_k(m)$ where f_k is a PRF ($F_n = \{f_k : \{0,1\}^n \to \{0,1\}^n\}$), where $V_k(m,t) \in \{\text{ACC}, \text{REJ}\}$.

If $F = U_n F_n$ is a PRF family then the MAC described above is secure. Let F be a PPT forger.

$$\Pr[F^R \text{ forges}] \leq 2^{-n}$$

$$\Pr[F^{f_k} \text{ forges}] \leq 2^{-n} + \text{neg}$$

If such a forger were to exist, then it would be able to decide between truly random and pseudorandom.

3.4 Arbitrary length MAC

Given a (G, M, V) that is a fixed length MAC for a blocks of size n.

Attempt 1:

$$M'_k(m_1, \ldots, m_d) = M_k(m_1 \oplus \cdots \oplus m_d)$$

However, this is equivalent to:

$$M'_k(m_2, m_1, \ldots, m_d)$$

which is a message that was never sent.

Attempt 2:

$$M'_k(m_1, \ldots, m_d) = (M_k(m_1), \ldots, M_k(m_d))$$

However, this is also insecure because we can construct:

$$M'_k(m_2, m_1, \ldots, m_d) = (M_k(m_2), M_k(m_1), \ldots, M_k(m_d))$$

Attempt 3:

$$M'_k(m_1, \ldots, m_d) = M_k(m_1, 1), \ldots, M_k(m_d, d)$$

Where each block m_1 has size $n/2$.

However, if we take two different messages from the oracle, we can mix and match between the messages to construct a valid tag.

Attempt 4: Consider $M'_k(m)$ where $m = m_1, \ldots, m_d$ where $m_i \in \{0,1\}^{n/3}$. Let $r \xleftarrow{R} \{0,1\}^{n/3}$. Let $t_i \leftarrow M_k(m_i, i, d, r)$ Output $r, t_1, \ldots, t_d$

Let $V'_k(r, t_1, \ldots, t_d) = \text{ACC}$ iff $\forall i \in \{1, \ldots, d\}$ it holds that $V_k((m_i, i, d, r), t_i) = \text{ACC}$.

If (G, M, V) is secure then (G', M', V') is secure. Suppose that F' is a PPT that forges (G', M', V') with probability ε. F' gets MAC for polynomially many messages. If F' is successful then $m = m_1, \ldots, m_d$, $t = (r, t_1, \ldots, t_d)$.

$$\forall i \;\; V_k((m_i, i, d, r), t_i) = \text{ACC}$$

We have two cases. In the first case $t = (r, t_1, \ldots, t_d)$ contains a "new" r. This means that (m_i, i, d, r) was never previously authenticated with M_k. In the second case t does not contain a "new" r. Either r has appeared in only one previous MAC, in which case it would be equivalent to breaking M_k. The other possibility is that r has appeared in more than one previous MAC, which happens with probability $q^2/2^{n/3}$.

CBC MAC size of tag is n (not $dn + n/3$). Let $F = \{f_k : \{0,1\}^n \to \{0,1\}^n\}$ be a family of PRFs. Let $M_k(m_1, \ldots, m_d) = y_d$ where $y_i = f_k(m_i \oplus y_{i-1})$ and $y_0 = 0^n$.

CBC MAC is secure for $m \in \{0,1\}^{dn}$.

4 Digital Signatures

$$m^{sk}\ A^{vk}\ \xrightarrow{m, S_{sk}(m)}\ B\ \ V_{vk}(m, S_{sk}(m)) = \text{ACC}$$

which means that Alice signs a message using a signing key sk, sends that signature and the message to Bob, who knows the corresponding verification key vk who can then verify that the message was signed by sk.

This is publicly verifiable (vk) is not secret. It is transferable. It is also non-repudiable.

4.1 Syntax

A digital signature consists of three algorithms (G, S, V).

1. A key generator $(vk, sk) \xleftarrow{R} G(1^n)$
2. A signature algorithm $\sigma \xleftarrow{R} S_{sk}(m)$
3. a verification algorithm $V_{vk}(m, \sigma) \in \{\text{ACC}, \text{REJ}\}$

We require that $V_{vk}(m, S_{sk}(m)) = \text{ACC}$ for any message m.

Note that the sender is the one who needs sk, whereas in the PKE it is the receiver. However, these are **not** the dual of PKE. Decryption does not imply signing; notably, $x^2 \bmod n$. Randomization is not necessary.

Security is defined exactly as in MACs, but where the adversary already has the verification key.

4.2 Applications

1. Public key infrastructure requires trust roots who certify that a given key belongs to an entity. For example, Verisign will provide Alice with a certificate that her keys are actually hers. When someone begins communicating with her, they can verify that the party on the other end actually has Alice's keys, assuming they trust Verisign.
2. Blockchains are built on the concept that a transfer is signed by the originating keypair that something has been transferred.

4.3 Example constructions

Attempt 1 Let f be a OWF such that:

1. $y = f(x)$ where $sk = x$, $vk = y$, and $x \xleftarrow{R} \{0,1\}^n$
2. $S_{sk}(m) = x$
3. $V_{vk}(m, \sigma) = 1 \Leftrightarrow f(\sigma) = y$

Attempt 2 Let $F = \{f_i : D_i \to D_i\}$ be a family of TDPs[1].

1. $vk = i$, $sk = t$
2. $S_{sk}(m) = f^{-1}(m)$
3. $V_{vk}(m, \sigma)$ checks that $f_i(\sigma) = m$

However, we can attack this without seeing any messages by showing that for $m = 1$, $\sigma = 1$ if we choose RSA as the trapdoor function. Another attack is to choose some signature $\sigma \in D$ and then compute the corresponding message $m = f_i(\sigma)$.

Attempt 3 Textbook RSA.

1. $vk = (N, d)$, $sk = (N, e)$
2. $S_{sk}(m) = m^e \mod N$
3. $V_{pk}(m, \sigma) = 1 \Leftrightarrow \sigma^d = m \mod N$

However, note that $(m \cdot m')^e = m^e m'^e \mod N$. To forge, we just need to ask for σ_1, σ_2 corresponding to m_1, m_2 where $m_2 = m_1^{-1} \cdot m \mod N$. $\sigma_1 \cdot \sigma_2$ is a valid signature on $m_1 \cdot m_2 = m_1 \cdot m_1^{-1} \cdot m = m$.

[1] Trap-door-permutations

Attempt 4 As a variation on this, we can use what is called a full domain hash, which requires a hash function H that breaks the homomorphisms such as $(m \cdot m')^e = m^e m'^e \mod N$.

1. $vk = (N, d)$, $sk = (N, e)$
2. $S_{sk}(m) = H(m)^e \mod N$
3. $V_{pk}(m, \sigma) = 1 \Leftrightarrow \sigma^d = m \mod N$

5 Next lecture

References

[1] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition.* Chapman & Hall/CRC, 2nd edition, 2014.