EE 720: An Introduction to Number Theory and Cryptography (Spring 2019)

Lecture 11 — February 7, 2019

Instructor: Saravanan Vijayakumaran *Scribe: Saravanan Vijayakumaran*

1 Lecture Plan

- Define message authentication codes
- Construction and security proof of a fixed-length MAC

2 Recap

- *CCA indistinguishability experiment* $\mathtt{PrivK}^{\mathtt{cca}}_{\mathcal{A},\Pi}(n)$:
  1. A key k is generated by running $\mathtt{Gen}(1^n)$.
  2. The adversary $\mathcal{A}$ is given 1^n and oracle access to $\mathtt{Enc}_k(\cdot)$ and $\mathtt{Dec}_k(\cdot)$. It outputs a pair of messages $m_0, m_1 \in \mathcal{M}$ with $|m_0| = |m_1|$.
  3. A uniform bit $b \in \{0, 1\}$ is chosen. Ciphertext $c \leftarrow \mathtt{Enc}_k(m_b)$ is computed and given to $\mathcal{A}$. c is called the *challenge ciphertext*.
  4. The adversary $\mathcal{A}$ continues to have oracle access to $\mathtt{Enc}_k(\cdot)$ and $\mathtt{Dec}_k(\cdot)$, but is not allowed to query the latter on the challenge ciphertext itself. Eventually, A outputs a bit b'.
  5. The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. If output is 1, we say that $\mathcal{A}$ succeeds.

Definition. *A private-key encryption scheme $\Pi = (\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ has **indistinguishable encryptions under a chosen-ciphertext attack**, or is **CCA-secure**, if for all probabilistic polynomial-time adversaries $\mathcal{A}$ there is a negligible function $\mathtt{negl}$ such that, for all n,*

$$\Pr\left[\mathtt{PrivK}^{\mathtt{cca}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \mathtt{negl}(n).$$

- Existence of padding oracle attacks justifies the CCA-security formulation.

3 Message Authentication Codes

- The main goal of cryptography is enabling secure communication between parties over an open communication channel. In addition to message privacy, secure communication entails *message integrity or authentication.*
- Each party should be able to check that a message it receives was sent by the party claiming to send it and that it was not modified in transit.

- Consider a scenario when a bank receives a request to transfer amount N from account X to account Y.
  - Is the request authentic? Did the owner of account X really raise the request?
  - Assuming the request is authentic, are the details exactly as specified by the owner of account X? Was the transfer amount modified?
- Message authentication codes prevent *undetected tampering* of messages sent over an open communication channel.
- In general, encryption schemes do not ensure message integrity. For example, given $c := G(k) \oplus m$, where k is a secret key and G is a pseudorandom generator, flipping a bit in c will flip the corresponding bit in the decrypted plaintext.

3.1 The Syntax of a Message Authentication Code

- We will continue to assume that the communicating parties share a secret key.
- When Alice wants to send a message m to Bob, she computes a `MAC tag` t based on the message and the shared key. Let `Mac` denote the *tag-generation algorithm*. Alice computes tag $t \leftarrow \mathtt{Mac}_k(m)$ and send (m, t) to Bob.
- Upon receiving (m, t), Bob verifies that t is a valid tag on the message m using a *verification algorithm* `Vrfy` which depends on the shared key k. $\mathtt{Vrfy}_k(m, t) = 1$ if t is a valid tag for m and 0 otherwise.

Definition. *A* ***message authentication code*** *(MAC) consists of three PPT algorithms* $(\mathtt{Gen}, \mathtt{Mac}, \mathtt{Vrfy})$ *such that:*

1. *The* ***key-generation algorithm*** `Gen` *takes as input the security parameter* 1^n *and outputs a key* k *with* $|k| \geq n$.
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* $t \leftarrow \mathtt{Mac}_k(m)$.
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* $b := \mathtt{Vrfy}_k(m, t)$.

It is required that for every n*, every key* k *output by* $\mathtt{Gen}(1^n)$*, and every* $m \in \{0,1\}^*$*, it holds that* $\mathtt{Vrfy}_k(m, \mathtt{Mac}_k(m)) = 1$.

If there is a function l *such that for every* k *output by* $\mathtt{Gen}(1^n)$*, algorithm* $\mathtt{Mac}_k$ *is only defined for messages* $m \in \{0,1\}^{l(n)}$*, then we call the scheme a* ***fixed length MAC for messages of length*** $l(n)$.

- **Canonical verification**: For deterministic message authentication codes (i.e. where `Mac` is a deterministic algorithm), the canonical way to perform verification is to simply re-compute the tag and check for equality.

3.2 Security of Message Authentication Codes

- The intuitive idea behind the security definition is that no efficient adversary should be able to generate a valid tag on any "new" message that was not previously sent (with tag) by one of the communicating parties.
- Consider the following **message authentication experiment** $\texttt{Mac-forge}_{\mathcal{A},\Pi}(n)$:
  1. A key k is generated by running $\texttt{Gen}(1^n)$.
  2. The adversary $\mathcal{A}$ is given input 1^n and oracle access to $\texttt{Mac}_k(\cdot)$. The adversary eventually outputs (m, t). Let $\mathcal{Q}$ denote the set of all queries that $\mathcal{A}$ asked its oracle.
  3. $\mathcal{A}$ succeeds if and only if (1) $\texttt{Vrfy}_k(m, t) = 1$ and (2) $m \notin \mathcal{Q}$. If $\mathcal{A}$ succeeds, the output of the experiment is 1. Otherwise, the output is 0.
- A MAC is secure if no efficient adversary can succeed in the above experiment with non-negligible probability.

Definition. *A message authentication code* $\Pi = (\texttt{Gen}, \texttt{Mac}, \texttt{Vrfy})$ *is* ***existentially unforgeable under an adaptive chosen-message attack****, or just* ***secure****, if for all PPT adversaries* $\mathcal{A}$*, there is a negligible function* $\texttt{negl}$ *such that:*

$$\Pr\left[\texttt{Mac-forge}_{\mathcal{A},\Pi}(n) = 1\right] \leq \texttt{negl}(n).$$

- The above definition of MAC security offers no protection against *replay attacks*. These can be prevented using sequence numbers or timestamps.

3.3 Fixed-Length MAC Construction

- Let F be a pseudorandom function. Define a fixed-length MAC for messages of length n as follows:
  - $\texttt{Mac}$: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, output the tag $t := F_k(m)$.
  - $\texttt{Vrfy}$: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, and a tag $t \in \{0,1\}^n$, output a 1 if and only if $t = F_k(m)$. If $t \neq F_k(m)$, output 0.

Theorem 1. *If F is a pseudorandom function, then the above construction is a secure fixed-length MAC for messages of length n.*

Proof. See pages 117–118 in Katz/Lindell. □

4 References and Additional Reading

- Sections 4.1, 4.2, 4.3 from Katz/Lindell